DRF CODES: DEEP SNR-ROBUST FEEDBACK CODES

Mahdi Boloursaz Mashhadi[1], Deniz Gündüz[2], Alberto Perotti[3], Branislav M. Popovic[3]
[1]5GIC & 6GIC, Institute for Communication Systems (ICS), University of Surrey, UK, [2]Dept. of Electrical and Electronic Engineering, Imperial College London, UK, [3]Radio Transmission Technology Laboratory, Huawei Technologies Sweden AB, Sweden

NOTE: Corresponding author: Mahdi Boloursaz Mashhadi, m.boloursazmashhadi@surrey.ac.uk

Abstract *– We present a new Deep Neural Network (DNN)-based error correction code for fading channels with output feedback, called the Deep SNR-Robust Feedback (DRF) code. At the encoder, parity symbols are generated by a Long Short Term Memory (LSTM) network based on the message, as well as the past forward channel outputs observed by the transmitter in a noisy fashion. The decoder uses a bidirectional LSTM architecture along with a Signal to Noise Ratio (SNR)-aware attention NN to decode the message. The proposed code overcomes two major shortcomings of DNN-based codes over channels with passive output feedback: (i) the SNR-aware attention mechanism at the decoder enables reliable application of the same trained NN over a wide range of SNR values; (ii) curriculum training with batch size scheduling is used to speed up and stabilize training while improving the SNR-robustness of the resulting code. We show that the DRF codes outperform the existing DNN-based codes in terms of both the SNR-robustness and the error rate in an Additive White Gaussian Noise (AWGN) channel with noisy output feedback. In fading channels with perfect phase compensation at the receiver, DRF codes learn to efficiently exploit knowledge of the instantaneous fading amplitude (which is available to the encoder through feedback) to reduce the overhead and complexity associated with channel estimation at the decoder. Finally, we show the effectiveness of DRF codes in multicast channels with feedback, where linear feedback codes are known to be strictly suboptimal. These results show the feasibility of automatic design of new channel codes using DNN-based language models.*

Keywords – Attention neural networks, channel coding, communication with feedback, curriculum training, LSTM language models

1. INTRODUCTION

Most wireless communication systems incorporate some form of feedback from the receiver to the transmitter. Retransmission mechanisms, such as Hybrid Automatic Repeat request (HARQ), or Channel State Information (CSI) feedback mechanisms in time-varying channels are common forms of feedback in wireless communications. Perfect channel output feedback can significantly boost communications' reliability at finite block lengths [1, 2, 3, 4], but does not improve the Shannon capacity of the channel [5]. Codes that make full use of feedback can potentially achieve improved performance as predicted in [4]. However, the design of reliable codes for channels with feedback has been a long-standing and notoriously difficult open problem. Several coding schemes for channels with feedback have been proposed [6, 1, 7, 8, 9, 10]; however, known solutions either do not approach the performance predicted in [4], or introduce unaffordable complexity. These schemes are also extremely sensitive to both the precision of the numerical computations and the noise in the feedback channel [2, 3]. It has been proven that with noisy output feedback, linear coding schemes fail to achieve any positive rate[11]. This is especially troubling since all practical codes are linear and linear codes are known to achieve capacity without feedback [12], and boost the error performance significantly in the case of noiseless feedback [1]. For the noisy feedback case, considerable improvements have been achieved using nonlinear modulo operations [13, 14].

More recently, some progress has been made by applying Machine Learning (ML) techniques, where channel decoding is regarded as a classification task, and the encoder and decoder, implemented as Deep Neural Network (DNN) architectures, are jointly trained in a data-driven fashion [15, 16, 17, 18, 19, 20, 21]. In this context, the encoder/decoder pair forms an over-complete autoencoder, where the encoder DNN adds redundancy to the latent representation of the message to cope with the channel noise, and the decoder DNN extracts features from the noisy received signal for efficient classification. In [15], the authors propose Deepcode for channels with noisy passive output feedback, consisting of a Recurrent Neural Network (RNN) encoder architecture along with a two-layer bidirectional Gated Recurrent Unit (GRU) decoder architecture, which are trained jointly on a dataset of random input/output realizations of the channel. In [16], a Convolutional Neural Network (CNN) encoder/decoder architecture with interleaving is used. In this work, the authors introduce interleaving to enable the CNN-based code to achieve a block length gain (i.e., decaying probability of error as the block length increases). In [17], Deep Extended Feedback (DEF) codes are introduced, which improve the error correction capability in comparison with Deepcode by an extended feedback mechanism that introduces longer range dependencies within the code blocks. DEF codes also enable higher spectral efficiencies by introducing higher order modulations into the encoding process. After the initial introduction of the DRF codes [22], other codes with more complex DNN architectures were proposed for the feedback channel. In particular, in [18] the authors proposed AttentionCode based on self-attention and restructuring, for reliable short-packet communications. In [19], the Generalized Block Attention Feedback (GBAF) codes were proposed based on the transformer architecture. GBAF codes utilize block-by-block processing to reduce the communication overhead, due to fewer interactions between the transmitter and receiver, and enable flexible coding rates. In [20], the authors extend GBAF codes to the active coded feedback scenario by implementing a pair of transformer architectures, at the transmitter and the receiver, which interact with each other sequentially. The transformer-based designs AttentionCode and GBAF [18, 19] outperform RNN and Long Short Term Memory (LSTM)-based codes [15, 17] in terms of the Block Error Rate (BLER) for channels with noiseless feedback or when the feedback Signal to Noise Ratio (SNR) is high, typically 20 dB and above.

Despite their significant performance, DNN-based codes are very sensitive to the mismatch between the actual channel SNR and the SNR that the NN has been trained for, which limits their application in practical communication systems with time-varying SNR values. Although similar performance degradation is also observed with traditional channel codes when there is a mismatch between the actual channel SNR and the SNR estimation used for decoding (e.g., due to an SNR estimation error) [23, 24], the impact is more critical for DNN-based codes. Since for DNN-based codes, the encoder and decoder are trained jointly, not only the decoder but also the transmitted codewords depend on the SNR. Hence, for practical deployment of the DNN-based codes on a time-varying channel, the encoder and decoder will have to train and store distinct DNNs for different SNR values, and use the appropriate one for each instantaneous value of the channel SNR, or more practically, for different ranges of SNR. This significantly increases the memory requirements and limits the practical application of DNN-based codes on realistic systems, and is a main focus of this paper.

In this paper, we propose Deep SNR-Robust Feedback (DRF) codes for the fading channels with noisy output feedback, which overcome the above-mentioned limitation of DNN-based channel codes. The DRF encoder transmits a message followed by a sequence of parity symbols, which are generated by an LSTM architecture based on the message, as well as the delayed past forward channel outputs observed by the encoder through a noisy feedback channel. The decoder uses a bidirectional LSTM architecture along with an SNR-aware attention [25, 26, 27, 28, 29] network to decode the message. In comparison with the transformer-based codes [18, 19], the LSTM-based DRF and DEF codes achieve lower BLER for lower feedback channel SNR ranges (i.e., feedback SNR 0-10 dB). LSTM-based designs also require less computational complexity for the same block length in comparison with their transformer-based counterparts [18]. The major contributions of this paper can be summarized as follows:

- We propose an attention mechanism that enables SNR-aware decoding of the DRF code, thereby considerably improving its robustness in realistic time-varying channels, where there may be a considerable mismatch between the training SNR and the instantaneous channel SNR. The attention module takes as input the forward and feedback noise variances and outputs attention coefficients that scale the features extracted at the output of the bidirectional LSTM layers at the decoder. The SNR-aware attention module significantly improves robustness of the code to a mismatch between the training and link-level SNR values for the AWGN channel with passive output feedback.

- We propose a training approach with SNR scheduling and batch-size adaptation. We start the training at low SNR values and with a smaller batch size, and gradually increase the SNR and the batch size along the training epochs according to a schedule. The proposed training approach improves the SNR-robustness of the resulting code and speeds up the training. The DRF codes and the proposed training approach not only achieve considerable SNR-robustness, but also improve the error rate over Deepcode [15] roughly by an order of magnitude. In comparison with the transformer-based AttentionCode [18], we show that the DRF code achieves lower BLER for lower feedback channel SNR ranges (i.e., feedback SNR 0-9 dB).

- For fading channels, in which the instantaneous SNR may be varying on each transmitted codeword (slow fading) or symbol (fast fading), we show that the proposed DRF codes learn to efficiently exploit the CSI, which is available to the encoder through feedback, such that no further improvement is possible by providing the CSI to the decoder. This is a desirable feature as it means that the complexity and overhead associated with channel estimation at the decoder can be reduced.

- For AWGN multicast channels with feedback, we show the power of DRF codes in exploiting multiple feedback signals to improve the reliability of all the receivers simultaneously. This is of significant interest as linear-feedback schemes are known to be strictly suboptimal for such channels [30].

The rest of this paper is organized as follows. In Section 2, we present the feedback channel model considered in this paper. In Section 3, we provide the DNN architectures for the DRF encoder and decoder.

In Section 4, we present our proposed training technique. Section 5 presents the simulation results. Section 6 extends the DRF codes to multicast channels with feedback, and Section 7 concludes the paper. ***Notations:*** Throughout this paper, we denote matrices and vectors by boldface lowercase and uppercase letters, respectively. All vectors are assumed to be column vectors. The notations $(\cdot)^T$ and $(\cdot)^{-1}$ are used for matrix transposition and inversion, respectively. Calligraphic letters denote sets, where $|\cdot|$ denotes cardinality of the set. Moreover, we denote the gradient by $\nabla(\cdot)$. Finally, $Pr\{\cdot\}$ denotes probability of an event, and $\mathbb{E}[\cdot]$ and var$(\cdot)$ denote the expectation and variance of random variables.

2. SYSTEM MODEL

Fig. 1 illustrates the canonical fading channel passive noisy output feedback that will be the focus of this research. Perfect phase compensation at the receiver is assumed and all variables are real-valued. In this model, we have:

$$y_i = \alpha_i x_i + n_i, \tag{1}$$

where x_i and y_i denote the channel input and output symbols, respectively, α_i is the channel fading coefficient, n_i is an independent and identically distributed (i.i.d.) Gaussian noise term, i.e., $n_i \sim \mathcal{N}(0, \sigma_n^2)$. We will assume that the channel fading coefficient comes from a prescribed distribution. We consider both slow and fast fading scenarios where the fading coefficient remains constant on each codeword for slow fading but gets i.i.d. random values on each symbol for the fast fading case. The channel output is assumed to be available at the encoder with a unit time delay via an independent AWGN feedback channel. At time i, the encoder has a noisy view of what was received at the decoder (in the past by one unit time):

$$z_i = y_{i-1} + m_i, \tag{2}$$

where m_i is an i.i.d. Gaussian noise term, i.e., $m_i \sim \mathcal{N}(0, \sigma_m^2)$. We call this a *passive* output feedback, as unlike in [13, 14], the decoder cannot apply any coding or other type of transformation on its received signal y_i before feeding it back to the encoder. The encoder can use the feedback symbol to sequentially and adaptively decide what to transmit as the next symbol. Therefore, channel input x_i at time instant i depends not only on the message $\mathbf{b} \in \{0,1\}^K$, but also on the past feedback symbols. The encoder maps the message $\mathbf{b} \in \{0,1\}^K$ onto the codeword $\mathbf{x} = [x_1, \dots, x_L]^T$, where L is the block length and K is the message length. The decoder maps the received codeword $\mathbf{y} = [y_1, \dots, y_L]^T$ into the estimated information bit sequence $\hat{\mathbf{b}} \in \{0,1\}^K$., where $r = K/L$ is the rate of the code. The BLER is given by BLER $= Pr\{\hat{\mathbf{b}} \neq \mathbf{b}\}$. Similarly, the Bit Error Rate (BER) is given by BER $= 1/K \sum_{k=1}^{K} Pr\{\hat{b}_k \neq b_k\}$, where b_k and $\hat{b}_k$ denote the k'th bit of the transmitted and reconstructed messages, respectively. We assume an average power constraint on the channel input, i.e., $\frac{1}{L}\mathbb{E}[\|\mathbf{x}\|^2] \leq 1$, where

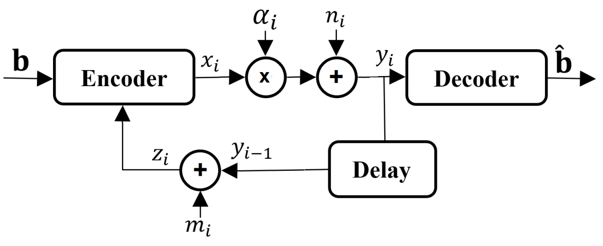


Fig. 1 – Fading channel with noisy *passive* output feedback

the expectation is over the randomness in the information bits, the randomness in the noisy feedback symbols $[z_1, \dots, z_L]^T$ and any other randomness in the encoder. We denote the forward and feedback channel SNR values by ρ and η, respectively, where:

$$\rho = 1/\sigma_n^2, \qquad \eta = 1/\sigma_m^2. \tag{3}$$

It should be noted that, we can generally identify two types of feedback mechanisms in communication systems, *Active* and *Passive* feedback. Here, we have considered the passive output feedback. Unlike active feedback, in the *Passive* feedback mechanism, the receiver does not perform any active processing on the received symbols prior to feedback. It is observed that when the SNR of the feedback link is considerably higher than the SNR of the forward channel, i.e. $\eta >> \rho$, any active processing at the receiver can be moved to the transmitter thanks to the high quality feedback link [18], thereby offloading the receiver by the corresponding computational effort. An example of a feedback channel with passive output feedback is the satellite-ground communications: the ground-to-satellite direction has a much larger transmission power than the reverse link, thereby providing high quality output feedback [1].

3. ENCODER/DECODER ARCHITECTURES

A major limitation of the existing DNN-based code designs in [15, 16, 17] is their dependencies on the channel SNR. That is, the encoder-decoder pairs are trained jointly for a specific SNR value. This means that, to be able to use these codes in practice, we will have to train and store a different DNN pair for different ranges of SNR values, which significantly limits their practical use in realistic channels with varying SNR. On the other hand, in conventional channel codes, the encoder depends only on the transmit power constraint, and the decoder uses the same decoding algorithm for all SNR values after converting the channel outputs into likelihood values depending on the channel SNR. Accordingly, a major goal of our paper is to implement a similar approach for DNN-based code design. This is achieved in this paper by incorporating an attention mechanism into the decoder of our proposed DRF code. This will allow us to train and store a single DNN, which can be used for all SNR values. Apart from this, we design the DRF code for fading channels with feedback, when the instantaneous channel SNR may change over time. Fig. 2 depicts our proposed DRF encoder and decoder architectures for a rate $50/153$ code.

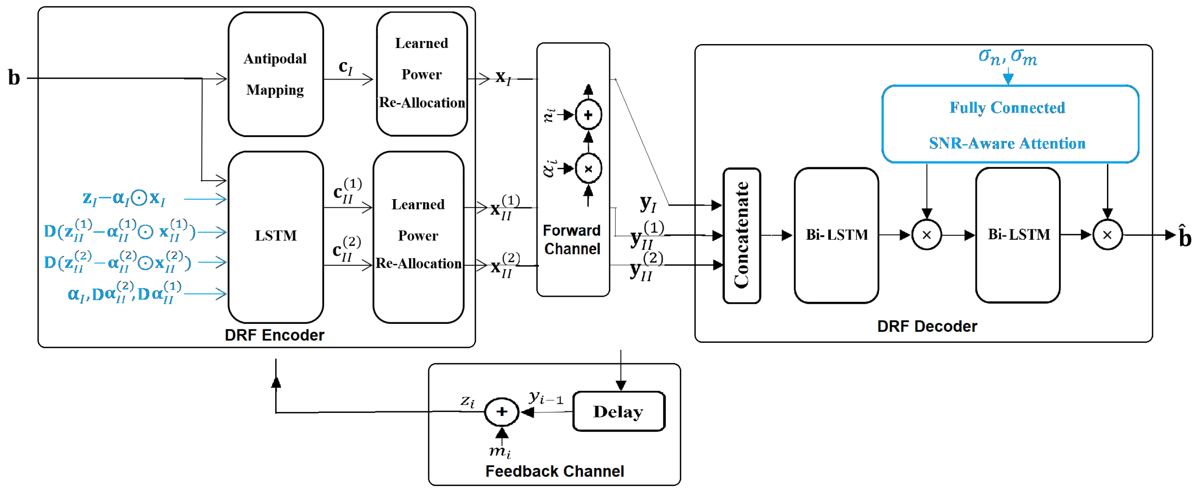


Fig. 2 – The block diagram of the proposed DRF code structure. The novel blocks of the DRF encoder and decoder architecture are shown in blue for emphasis

3.1 Encoder

Fig. 2 illustrates the encoder architecture. Encoding is a two-phase process: in phase I, the vector $\mathbf{b} = [b_1, \dots, b_K, 0]^T$ consisting of the message bits padded by a zero is transmitted over the channel by an antipodal mapping, i.e., $\mathbf{c}_I = 2\mathbf{b} - 1$. Zero padding is applied to mitigate the increasing error rate effects on the last few bits of the block as suggested in [15]. During phase II, the encoder uses a 1-layer LSTM [31] network, including $K+1$ LSTM units to generate two sets of parity symbols, i.e., $\mathbf{c}_{II}^{(1)}$ and $\mathbf{c}_{II}^{(2)}$, based on the observations of channel noise and fading in phase I and the delayed noise and fading in phase II on each of the two sets of parity symbols. We use single directional LSTM units due to the causality constraint enforced by the channel model. The LSTM activation is hyperbolic tangent, i.e., $\tanh(x) = \frac{e^x-e^{-x}}{e^x+e^{-x}}$, while the output activation function is sigmoid, i.e., $\text{sigmoid}(x) = \frac{1}{1+e^{-x}}$. The resulting code block transmitted over the channel is $\mathbf{x} = [\mathbf{x}_I^T, \mathbf{x}_{II}^{(1)T}, \mathbf{x}_{II}^{(2)T}]^T = [x_1, \dots, x_{3K+3}]^T$, where $\mathbf{x}_I = \mathcal{P}\{\mathbf{c}_I\} = [x_1, x_2, \dots, x_{K+1}]^T$, $\mathbf{x}_{II}^{(1)} = \mathcal{P}\{\mathbf{c}_{II}^{(1)}\} = [x_{K+2}, x_{K+3}, \dots, x_{2K+2}]^T$, and $\mathbf{x}_{II}^{(2)} = \mathcal{P}\{\mathbf{c}_{II}^{(2)}\} = [x_{2K+3}, x_{2K+4}, \dots, x_{3K+3}]^T$. Here, $\mathcal{P}\{\cdot\}$ denotes a learned power reallocation layer to balance the error over the whole block as suggested in [15]. Please note that we use antipodal, i.e. BPSK, signaling in phase I transmission but the parity symbols generated in phase II are discrete-time continuous amplitude symbols that can take arbitrary real values. These continuous amplitude symbols are then directly mapped to the channel input, e.g. OFDM grid resources, following the channel input power constraint.

The encoder estimates the forward channel from observations of the feedback. The encoder knows the transmitted symbol x_i and also observes the corresponding feedback symbol $z_i = \alpha_{i-1}x_{i-1} + n_{i-1} + m_i$ with a single delay, and therefore, it can estimate the CSI $\hat{\alpha}_{i-1}$ from its observation. This estimate of the CSI is then input to the encoder. For example, in a fast fading scenario, where the fading coefficient takes random i.i.d. realizations on each symbol, the Linear Minimum Mean Square Error (LMMSE) estimate of the channel gain can be calculated by:

$$\hat{\alpha}_i = \frac{x_{i-1}\text{var}(\alpha)}{|x_{i-1}|^2\text{var}(\alpha) + \sigma_n^2 + \sigma_m^2} z_i + \frac{\sigma_n^2 + \sigma_m^2}{|x_{i-1}|^2\text{var}(\alpha) + \sigma_n^2 + \sigma_m^2}\mathbb{E}[\alpha], \tag{4}$$

where $\mathbb{E}[\alpha]$ and $\text{var}(\alpha)$ denote the expected value and variance of the fading coefficient, respectively. In a slow fading scenario, the fading coefficient is fixed over the whole codeword, i.e., for the considered rate $50/153$ code with a single bit zero padding we have $\alpha_1 = \cdots = \alpha_{3K+3} = \alpha$. The fading coefficient α takes random i.i.d. realizations over different codewords, and the transmitter uses the causal vectors $\mathbf{z}_i = [z_1, \dots, z_i]^T$ and $\mathbf{x}_i = [x_1, \dots, x_i]^T$, to calculate the LMMSE channel estimate as

$$\hat{\alpha} = \text{var}(\alpha)\mathbf{x}_{i-1}^T(\text{var}(\alpha)\mathbf{x}_{i-1}\mathbf{x}_{i-1}^T + \sigma_n^2 I + \sigma_m^2 I)^{-1}\mathbf{z}_i + \mathbb{E}[\alpha](1 - \text{var}(\alpha)\mathbf{x}_{i-1}^T(\text{var}(\alpha)\mathbf{x}_{i-1}\mathbf{x}_{i-1}^T + \sigma_n^2 I + \sigma_m^2 I)^{-1}\mathbf{x}_{i-1}), \tag{5}$$

in which I is the identity matrix. In these equations, knowledge of $\mathbb{E}[\alpha]$ and $\text{var}(\alpha)$ at the transmitter is assumed.

The causal CSI available at the encoder is fed into the LSTM units to cope with channel uncertainty due to fading. To this end, we concatenate the vector of instantaneous channel fading coefficients in phase I, i.e. $\boldsymbol{\alpha}_I = [\alpha_1, \dots, \alpha_{K+1}]^T$, and the causal fading coefficient in phase II i.e. $\mathbf{D}\boldsymbol{\alpha}_{II}^{(1)} = [0, \alpha_{K+2}, \dots, \alpha_{2K+1}]^T$, and $\mathbf{D}\boldsymbol{\alpha}_{II}^{(2)} = [0, \alpha_{2K+3}, \dots, \alpha_{3K+2}]^T$, and feed into the LSTM units at the encoder ($\mathbf{D}$ denotes a single delay, see Fig. 2). We also provide estimates of the noise in the forward and feedback channels to the encoder, i.e. $\mathbf{z}_I - \alpha_I \odot \mathbf{x}_I$,

$\mathbf{D}(\mathbf{z}_{II}^{(1)} - \alpha_{II}^{(1)} \odot \mathbf{x}_{II}^{(1)})$, and $\mathbf{D}(\mathbf{z}_{II}^{(2)} - \alpha_{II}^{(2)} \odot \mathbf{x}_{II}^{(2)})$, where $\odot$ denotes element-wise multiplication. For the AWGN case where $\boldsymbol{\alpha}_I = \boldsymbol{\alpha}_{II}^{(1)} = \boldsymbol{\alpha}_{II}^{(2)} = 1$, the corresponding inputs are omitted to avoid unnecessary complexity.

3.2 Decoder

Fig. 2 illustrates the DRF decoder consisting of a two-layer LSTM architecture (each including $K+1$ LSTM units) and an SNR-aware fully connected attention network used for feature scaling. At the decoder, we use bidirectional LSTM layers to exploit long range forward and backward dependencies in the received code block. The phase I and II received signals are concatenated at the decoder and fed to the bidirectional LSTM layers. Each LSTM layer is followed by batch normalization. In a similar way to the encoder, the LSTM activation is hyperbolic tangent while the output activation is sigmoid. The bidirectional LSTM layers extract features from the noisy received signals, which are then used for efficient decoding.
Note that we use LSTM layers at both the encoder and the decoder, which, according to our observations, considerably reduce the error rate in comparison with simple RNN and Gated Recurrent Unit (GRU) layers used in [15]. This is because LSTM layers can better learn long-range dependencies by avoiding the gradient vanishing problem in training long RNN layers [32, 33]. The LSTM architecture includes a set of gates that control when longer range information enters the memory, when it is output, and when it is forgotten [31]. This property is very favourable for channel encoding and decoding as generating redundancies based on long range dependencies is essential to achieving a block length gain.

3.3 SNR-aware attention

A major novelty in our decoder architecture is the SNR-aware attention module. An attention mechanism is a vector of importance weights to measure the correlations between a vector of inputs and the target to be predicted. Attention weights are calculated as a parameterized attention function with learnable parameters [25, 26, 27, 28, 29]. We use a two-layer Fully Connected (FC) attention at the DRF decoder as outlined in Table 1. The idea is to let the attention layers learn how much each bi-LSTM output should be weighted according to the SNR. Also, by means of the attention module, we explicitly provide the noise standard deviation to the decoder, which enables learning codes that are capable of adaptation to the channel SNR, which in turn allows the use of the same trained encoder/decoder weights over a wide range of channel SNR values. Here, the standard deviations of the forward and feedback channel noise are obtained through link-level estimation. The number of attention weights determines the number of neurons at the last FC layer in Table 1 and equals $2HK$, where H is the length of the LSTM hidden state (i.e., $H = K$ here) and is multiplied by 2 because the LSTM layer is bidirectional.

The total number of FC attention layers and the number of neurons in each intermediate layer are hyperparameters optimized numerically for the best performance.

Table 1 – Model architecture for the SNR-aware attention module at the decoder

Layer	Output Dim.
Input	2
Fully connected + sigmoid	$4K^2$
Fully connected + sigmoid	$2K^2$

4. TRAINING DRF CODES

We denote the i'th training sample by $\mathbf{S}_i = \{\mathbf{b}_i, \boldsymbol{\alpha}_i, \mathbf{n}_i, \mathbf{m}_i\}$, which consists of a random realization of a message, i.e., $\mathbf{b}_i$, the corresponding realization of the channel fading coefficient $\boldsymbol{\alpha}_i$, and the forward and feedback noise realizations, $\mathbf{n}_i$ and $\mathbf{m}_i$, respectively. We denote the encoder and decoder functions by $f(\cdot;\boldsymbol{\theta})$ and $g(\cdot;\boldsymbol{\psi})$, where $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ are the trainable encoder and decoder parameters. We have, $\hat{\boldsymbol{b}}_i = g(\boldsymbol{\alpha}_i f(\mathbf{S}_i;\boldsymbol{\theta}) + \mathbf{n}_i;\boldsymbol{\psi})$. To train the model, we minimize

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\psi}, \mathcal{B}) = -\frac{1}{|\mathcal{B}|} \sum_{\mathbf{S}_i \in \mathcal{B}} l(\hat{\boldsymbol{b}}_i, \mathbf{b}_i; \boldsymbol{\theta}, \boldsymbol{\psi}), \tag{6}$$

where $\mathcal{B}$ is a batch of samples, $l(\hat{\boldsymbol{b}}_i, \mathbf{b}_i; \boldsymbol{\theta}, \boldsymbol{\psi})$ is the binary cross-entropy loss given by

$$l(\hat{\boldsymbol{b}}_i, \mathbf{b}_i; \boldsymbol{\theta}, \boldsymbol{\psi}) = \sum_{k=1}^{K} [\mathbf{b}_i]_k \log_2(1-[\hat{\boldsymbol{b}}_i]_k) + (1-[\mathbf{b}_i]_k) \log_2([\hat{\boldsymbol{b}}_i]_k), \tag{7}$$

and $[\mathbf{b}_i]_k$ and $[\hat{\boldsymbol{b}}_i]_k$ denote the kth bit of the message and its estimate.
To train the model, we use a variant of the Stochastic Gradient Descent (SGD), for which the vector of all trainable parameters $\boldsymbol{\phi}^T = [\boldsymbol{\theta}^T, \boldsymbol{\psi}^T]$ is updated by iterations of the form:

$$\boldsymbol{\phi}^{(t)} = \boldsymbol{\phi}^{(t-1)} - \mu_t \nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}^{(t-1)}, \mathcal{B}^{(t)}), \tag{8}$$

where t is the iteration index, $\mu_t > 0$ is the learning rate, and $\mathcal{B}^{(t)}$ is a random batch from the dataset.
To ensure that the model is trained with many random realizations of the data and noise, we generate and use a new random set of samples in each epoch. We denote the dataset used in the u'th training epoch by $\mathcal{D}^u = \{\mathbf{S}_i\}_{i=1}^{|\mathcal{D}^u|}$, where $|\mathcal{D}^u| = \zeta|\mathcal{B}^u|$, ζ is a constant and $|\mathcal{B}^u|$ is the batch size for the u'th epoch. Training DNNs with SGD, or its variants, requires careful choice of the training parameters (e.g., learning rate, batch size, etc.). For the specific task of training an efficient SNR-robust channel encoder and decoder, the SNR used to generate the training samples $\mathbf{S}_i$ also becomes a crucial parameter. In the following, we present our proposed training approach with SNR and batch size scheduling which enables faster training of the DRF codes while resulting in a reliable and SNR-robust encoder/decoder pair.

Algorithm 1: Training of DRF codes with batch size adaptation and SNR scheduling

```
   Input: $U, \rho_1 \leq \rho_2 \leq \cdots \leq \rho_U, |\mathcal{B}^1|, B_{max}, \zeta, \lambda, \kappa$
   Data: $\mathcal{L}_0 = \infty$
1  for epoch $u = 1, 2, \ldots, U$ do
2      Randomly generate training dataset $\mathcal{D}^u$
         consisting of
3      $\zeta|\mathcal{B}^u|$ samples with forward SNR $\rho_u$
4      Perform one epoch of training using SGD as in
       (8) and record final loss $\mathcal{L}_u$
5      if $(\mathcal{L}_u \geq \lambda\mathcal{L}_{u-1})\&(|\mathcal{B}^u| < B_{max})$ then
6          Update batch size $|\mathcal{B}^{u+1}| = \kappa|\mathcal{B}^u|$
7      else
8          $|\mathcal{B}^{u+1}| = |\mathcal{B}^u|$
9      end
10 end
   Output: Trained encoder/decoder parameters $\boldsymbol{\theta}, \boldsymbol{\psi}$
```

4.1 Batch-size adaptation

In training machine learning models, a static batch size held constant throughout the training process forces the user to resolve a tradeoff. On one hand, small batch sizes are desirable since they tend to achieve faster convergence. On the other hand, large batch sizes offer more data-parallelism, which in turn improves computational efficiency and scalability [34, 35]. However, for the specific channel encoder/decoder training task a significantly larger batch size is necessary not only due to the data-parallelism benefits, but also because after a few training steps, the error rate and consequently the binary cross-entropy loss *(6)* becomes very small, typically $10^{-4} \sim 10^{-7}$ for the range of SNR values considered here. Hence, to get a statistically accurate estimate of such a small loss value, and consequently, an accurate estimate of the gradient update in *(8)*, the batch size must be very large (typically ~ 10000 samples here).

If the batch size is small, the performance saturates after several epochs meaning that the optimizer enters a state when it keeps iterating with inaccurate gradient estimates leading to random fluctuations in the loss value with no actual improvement. This is due to the fact that a small batch size cannot provide an accurate estimate of the gradient, and hence, training is stuck oscillating around a minimum value, but is not able to further approach it. In this situation, as the batch size is not sufficiently large to estimate the gradient accurately, an unfortunate random realization of any batch can lead to a destructive update of the model parameters causing sudden jumps in the loss function and the BLER along the training. It was observed in simulations that such destructive updates hamper convergence, and at times, can totally destroy the code and result in divergence.

To avoid these destructive updates, we here propose an adaptive batch size scheme tailored for training a DNN-based channel encoder and decoder pair. In this scheme, we train the model starting from a small batch size $|\mathcal{B}^1|$, and multiply the batch size by a factor of κ whenever the cross-entropy loss does not decrease by a factor of λ in two consecutive epochs, until we reach a maximum batch size value B_{max}. The maximum batch size is constrained by the memory resources available to our training platform. We hence train with a sequence of batch sizes, $|\mathcal{B}^1| \leq |\mathcal{B}^2| \leq \cdots \leq |\mathcal{B}^U| \leq B_{max}$, where U is the total number of epochs. Starting from a smaller batch size enables a faster convergence during initial epochs. We increase the batch size whenever trapped around a minimum due to insufficiency of the batch size to achieve an accurate estimate of the gradient. This way, destructive updates become less likely as we avoid iterating with inaccurate gradient estimates. The proposed batch size adaptation stabilizes and speeds up the training process by avoiding destructive updates due to batch size insufficiency.

4.2 SNR scheduling

When training the channel encoder/decoder pair for a range of SNR values, if low and high SNR samples are presented to the decoder together during training, the trained NN tends to be biased towards the lower SNR. This is because the error probability for higher SNR values can be orders of magnitude smaller than the lower ones. Hence, the contribution of the high SNR samples in the batch to the binary cross-entropy loss *(6)* becomes negligible. In this case, the low SNR samples will decide the loss value and consequently the gradient updates *(8)* causing the channel code to be biased towards lower SNR values. On the other hand, training is easier for lower SNR values, in the sense that, for higher SNR values, destructive updates become more frequent causing the training to become less stable.

To train a channel encoder and decoder pair suitable for a wide SNR range, we here propose a scheduled-SNR training approach. This is motivated by the idea of curriculum training [36, 37], which suggests using a "curriculum" in presenting training samples to the DNN based on their "difficulty". Curriculum training improves both the speed of convergence of the training process, and the quality of the local minima obtained in the case of non-convex optimization criteria [36, 37].

Assume the goal is to efficiently train a channel encoder/decoder pair that works sufficiently well for all forward channel SNR values $\rho \in [\rho_{min}, \rho_{max}]$. We start training with lower SNR samples and increase the SNR along the epochs using an SNR schedule of $\rho_{min} = \rho_1 \leq \rho_2 \leq \cdots \leq \rho_U = \rho_{max}$. We observed that SNR scheduling combined with batch size adaptation not only stabilizes and speeds up the training, but also improves the SNR-robustness when training an encoder/decoder pair for a wider SNR range. Algorithm 1 summarizes our training approach for DRF codes. The hyperparameters U, $|\mathcal{B}^1|$, B_{max}, ζ, λ, and κ are chosen by numerical evaluations for the best performance.

Deployment protocols: The DRF codes are to be trained offline for various operational SNR ranges of the network and the corresponding code rates. The trained DRF code is assumed to be saved and available either in the UE itself or in a network entity closer to the UE, e.g. the corresponding Access Point (AP) or BS. Similar to the existing communication protocols, the UE and the AP or BS agree on an appropriate Modulation and Coding Scheme (MCS) [38] upon connection of the UE to the network, based on the channel conditions and available resources. After this, the corresponding trained DNN is used for communication which is either already available and saved at the UE, or cached into the UE from the AP or BS. If the channel conditions change significantly, then a new MCS will be agreed between the UE and AP or BS and the corresponding newly-trained DNN will be used (either locally available or cached). With this design, deployment of the DRF codes will have minimal effect on the existing communication protocols. Please note that a major benefit of our proposed DRF codes is to achieve significantly improved robustness to changes in the channel SNR in comparison with previous work, Deepcode [15]. This means that we need to train and save fewer DNNs, thereby saving both the time required for training the DNNs and the corresponding storage required to save the trained DNNs.
Computational complexity: Consider a (K, L) DRF code, the number of computations for execution of the DRF code is $\mathcal{O}(K\pi^2)$ where π is the size of LSTM latent variables, i.e., the number of latent features we use to represent one input. The training complexity is proportional to both the number of computations in the LSTM networks and the number of training epochs U. Hence, the training complexity of DRF codes is $\mathcal{O}(KU\pi^2)$. In simulations we have $K = 50, \pi = 50, U = 15$, and training of a single user DRF code takes roughly 3 hours on a NVIDIA GEFORCE RTX 28 Ti GPU. However, it is worth noting that the training of DRF codes is a one-shot cost. Once trained, the code can be deployed and only the execution complexity matters.
Storage requirements: Various MCS modes of operation in the existing wireless standards, e.g. [38], require different code rates. This will incur storage of various trained DNNs for each of the MCS values on the UE device. This is challenging for the UE as these trained DNNs are large, e.g. the DRF code has 87.9×10^6 float 32 parameters and the trained code takes $\sim$ 18MB on device when compressed. An alternative may be to cache and transmit these trained DNNs to the UE from the AP/BS during the initial connection establishment which will impose a communication overhead.

5. NUMERICAL EVALUATIONS

In this section, we evaluate the performance of the proposed DRF codes. In all simulations, we use 10^9 random samples to achieve a reliable estimate of the error rate. Each sample includes a random realization of the message **b**, and the corresponding random realizations of forward and feedback channels. In all the simulations, we set $K = 50, L = 153$, and the NN optimizer is Adam [39]. The values of the hyperparameters are: $U = 15$, $|\mathcal{B}^1| = 1000$, $B_{max} = 16000$, $\zeta = 100$, $\lambda = 2$, $\kappa = 2$.

5.1 AWGN channel

In this subsection, we consider the AWGN case, i.e. $\alpha_i = 1, \forall i$. We first show the robustness of the proposed DRF codes to a mismatch between the training and the actual channel SNR values. We then provide an ablation study to separately show the effectiveness of the SNR-aware attention mechanism and the proposed training approach to achieve SNR robustness. We finally provide BLER comparisons between the DRF codes and both the conventional and feedback channel codes. We show that DRF codes outperform the benchmark Low Density Parity Check (LDPC) codes adopted for the 5th Generation New Radio (5G NR) [4], by three orders of magnitude and the previously proposed Deepcode [15] by an order of magnitude.

5.1.1 SNR-robustness

We first compare the BLER of the proposed DRF codes with and without the attention module, when there is a mismatch between the actual channel SNR and the SNR used for training. The SNR mismatch is defined as $\Delta\rho = \rho - \hat{\rho}$ where ρ is the actual channel SNR and $\hat{\rho}$ is the SNR used for training.
The results are depicted in Fig. 3, where we plot the BLER versus $\Delta\rho$ for $\rho = -1, 0, 1$ dB. The results are plotted for both noiseless $\eta = \infty$ and noisy feedback at $\eta = 20$ dB. This figure shows that without the SNR-aware attention module at the decoder, the BLER is very sensitive to the SNR mismatch. In this case, a negative SNR mismatch (i.e., training SNR is higher than the actual channel SNR), can significantly degrade the BLER by orders of magnitude. The BLER is less sensitive to a positive mismatch but still roughly an order of magnitude BLER degradation is observed if there is $\Delta\rho = +3$ dB mismatch between the training and test SNR values. This figure shows that DRF codes are significantly more robust to both positive and negative SNR mismatch due to the SNR-aware attention layers added to the decoder.

5.1.2 Ablation study

In this subsection, we compare the BLER of the DRF architecture with and without SNR scheduling and attention, when the goal is to train a single DNN that works sufficiently well over all the SNR values $\rho \in \{-1, 0, 1, 2\}$ dB. The idea is to show the effectiveness of the proposed SNR-aware attention at the decoder and scheduling of the training SNR, separately. To this end, we report in Table 2, the BLER values versus channel SNR when the feedback link is noiseless. We train all the schemes with batch size adaptation. The results labeled as "DRF code without attention (Separate Trained DNNs)" in Table 2, shows the performance when we have trained four different DNNs

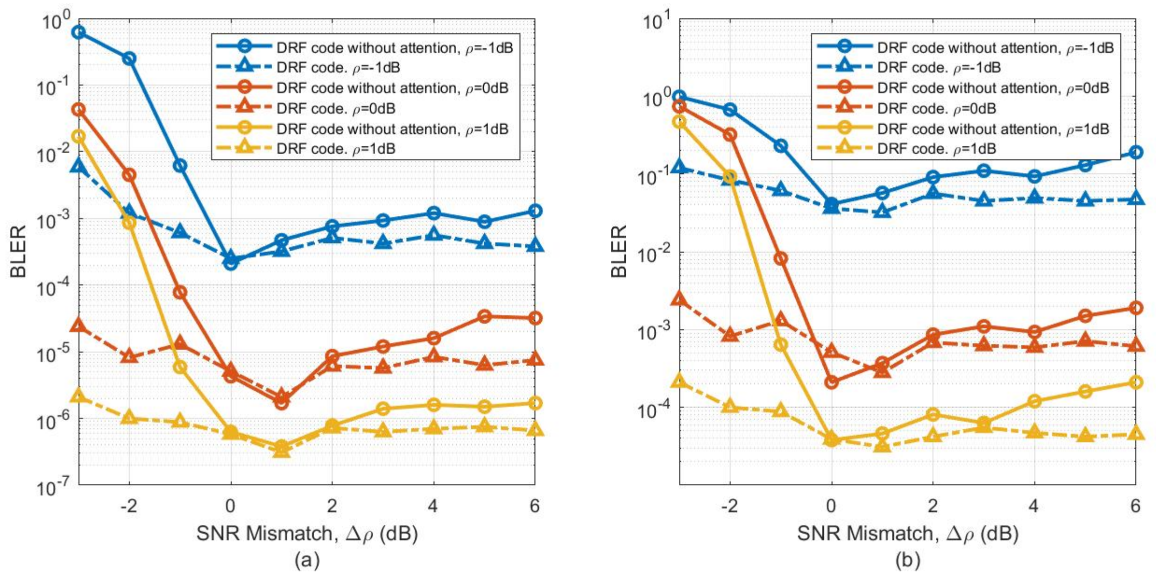


Fig. 3 – Comparison between the DRF code and LSTM-based Deepcode in terms of BLER as a function of SNR mismatch $\Delta\rho$. (a) Noiseless feedback ($\eta = \infty$), (b) Noisy feedback ($\eta = 20$ dB)

Table 2 – SNR robustness for the proposed DRF codes and comparison with Deepcode, noiseless feedback ($\eta = \infty$)

Test SNR	-1dB	0dB	1dB	2dB
DRF code without attention (Separate Trained DNNs)	2.1×10^{-4}	$\mathbf{1.7 \times 10^{-6}}$	3.8×10^{-7}	5.6×10^{-8}
DRF code without attention (Trained over $[-1, 2]$dB)	2.8×10^{-4}	7.9×10^{-6}	1.6×10^{-6}	4.8×10^{-7}
DRF code with attention (Trained over $[-1, 2]$dB)	$\mathbf{1.8 \times 10^{-4}}$	5.3×10^{-6}	9.7×10^{-7}	1.2×10^{-7}
DRF code (SNR Scheduling)	2.0×10^{-4}	2.4×10^{-6}	$\mathbf{2.9 \times 10^{-7}}$	$\mathbf{5.1 \times 10^{-8}}$

(without the attention module) at forward SNR values of $\rho \in \{-1, 0, 1, 2\}$ dB, and evaluated them on the same test SNR value. However, in a realistic time-varying channel, the instantaneous SNR varies with time (e.g., a slow channel fading scenario). In such cases, switching between separate DNNs for channel encoding/decoding is less practical.

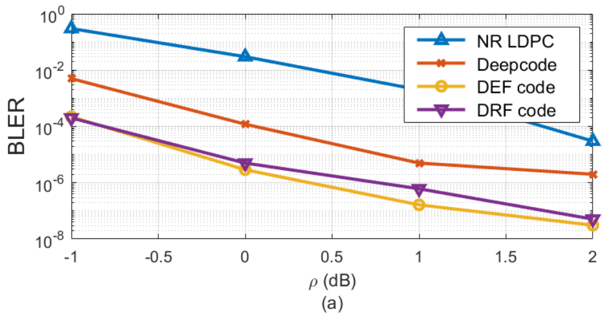
The results labeled as "DRF code without attention (Trained over $[-1, 2]$dB)" in Table 2, correspond to the scheme where we train a single DRF code architecture without attention on training samples generated with SNR values picked uniformly at random from $[-1, 2]$dB and then tested on each SNR value. As shown in Table 2, this approach leads to considerable performance degradation specifically at higher SNR values. This is because the trained code is considerably biased towards a better performance at low SNR values.

The results labeled as "DRF code with attention (Trained over $[-1, 2]$dB)" in Table 2, report the performance when a single DRF architecture (including the attention module) is trained on samples generated with SNR values picked uniformly at random from $[-1, 2]$dB and then tested on each SNR value. As the DRF decoder is aware of the SNR, it does not suffer as much performance degradation at high SNR values when it is trained over random SNR values. However, it is still slightly biased towards a better performance at low SNR values.

The results labeled as "DRF code (SNR Scheduling)" in Table 2, report the performance when a single DRF code architecture (including the attention module) is trained with the proposed SNR scheduling approach. Here, instead of training with samples generated with random SNR values picked from $[-1,\ 2]$dB, we train for three epochs on samples generated with each of the SNR values in the schedule "-1, -1, 0, 1, 2" dB, respectively, in that order (we observed in simulations that more training at SNR -1 dB improves the final performance). Comparing the last two rows of Table 2, we observe that curriculum training with an SNR schedule further improves the performance, specifically for the higher SNR values. According to these results, the proposed DRF code architecture along with SNR scheduling achieves BLER better than or comparable with the "Separate Trained DNNs" case while alleviating the need to train and store several DNNs for various SNR values, thereby, significantly improving practicality of the DNN-based code. Similar results are observed for the noisy feedback case with $\eta = 20$ dB.

5.1.3 Error rate comparisons

Fig. 4 compares the BLER values achieved for recurrent code designs over the AWGN channel with feedback, for forward channel SNR values in the range $\{-1, 2\}$ dB when (a) the feedback is noiseless ($\eta = \infty$), and (b) the feed-



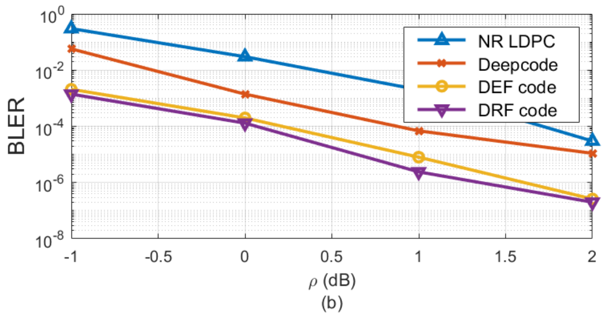


Fig. 4 – BER comparison between the recurrent code designs over AWGN channels with feedback. (a) Noiseless feedback ($\eta = \infty$), (b) Noisy feedback ($\eta = 20$ dB)

back SNR is $\eta = 20$ dB. Please note that the forward SNR range $[-1,\ 2]$ has been adopted in these simulations to stay consistent with the literature [15, 17, 18, 19, 20]. The SNR range $[-1, 2]$ is used in this line of research [15, 17, 18, 19, 20] due to the fact that the conventional forward error correction codes, e.g. NR LDPC, polar, turbo codes, perform poorly in this range; refer to Fig. 4 or [15], Fig. 2. Hence, better performing codes are required for the forward SNR values in the range $[-1, 2]$.

The orange curve reports the BLER for the RNN-based Deepcode architecture as proposed in [15]. According to this figure, the proposed DRF codes reduce the BLER by almost three orders of magnitude in comparison with NR LDPC and an order of magnitude in comparison with Deepcode [15]. Note that for the Deepcode and DEF code, we have trained and used a different DNN for each of the four SNR-BLER points corresponding to the markers on the curves of Fig. 4. However, for the DRF code, we have used a single DNN for all the SNR points, which is trained using our proposed SNR scheduling approach. In comparison with the DEF code, the DRF code achieves SNR-robustness with no performance degradation.

We note that although DNN-based codes achieve huge BLER reductions for the low SNR values considered here, we do not observe a decay as fast as the traditional channel codes (e.g., LDPC) in their error rate as the SNR increases. This may be due to the fact that at higher SNR values, the error rate and consequently the binary cross-entropy loss becomes too small to be accurately estimated with affordable batch size values making the training unstable as mentioned in Section 4. The error rate decay in the high SNR for the DNN-based codes will be further investigated in future research.

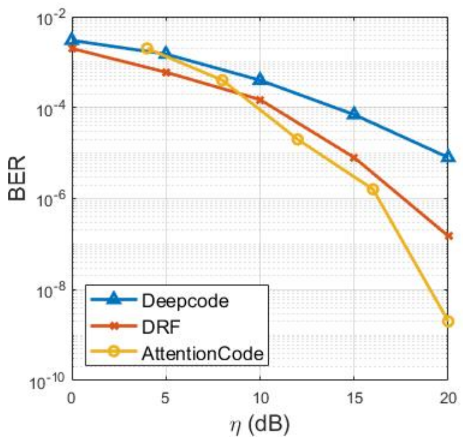


Fig. 5 – BER curves versus η for forward SNR $\rho = 1$ dB

In [19], the authors provide a comparison between recurrent and transformer-based codes over AWGN channels with feedback when the feedback channel is noiseless and the feedback SNR is 20 dB, and they show that for such channels, the transformer-based designs AttentionCode and GBAF [18, 19] achieve lower BLER in comparison with recurrent RNN and LSTM-based codes Deepcode, DEF, and DRF codes. Referring to [18], transformer-based designs require more computational complexity for the same block length in comparison with the recurrent DRF codes.DRF codes also achieve lower BER for lower feedback channel SNR ranges (i.e., feedback SNR 0-10 dB). In Fig. 5, we plot the BER versus feedback SNR curves when the forward SNR value is fixed at $\rho = 1$ dB. It is evident that DRF codes outperform Deepcode in a wide range of feedback channel SNR values, and achieve lower BER in comparison with the transformer based AttentionCode for feedback SNR lower than 9 dB.

5.2 Fading channel

In this subsection, we consider fading channels with feedback as depicted in Fig. 1. Depending on the wireless environment, the CSI coefficient α_i may follow various statistics [41, 42]. In this section we adopt the Rayleigh channel assumption, which is valid for rich scattering urban environments when there exists no dominant Line-of-Sight (LoS) multipath component. Hence, the forward channel gain α_i follows the probability density function (pdf) $f(\alpha) = \frac{\alpha}{\sigma^2} e^{\frac{-\alpha^2}{2\sigma^2}}, \alpha > 0$, where σ is the scale parameter and the average power gain of the fading channel is given by $\Omega = 2\sigma^2$.

We consider both slow and fast fading scenarios. In the slow fading case, the fading coefficient takes random i.i.d. Rayleigh realizations over each transmitted codeword, but remains constant throughout the transmission of the codeword. In the fast fading case, fading coefficients take random i.i.d. Rayleigh realizations on each transmitted symbol. This is an extreme case as we assume consecutive channel realizations to be independent, while these are

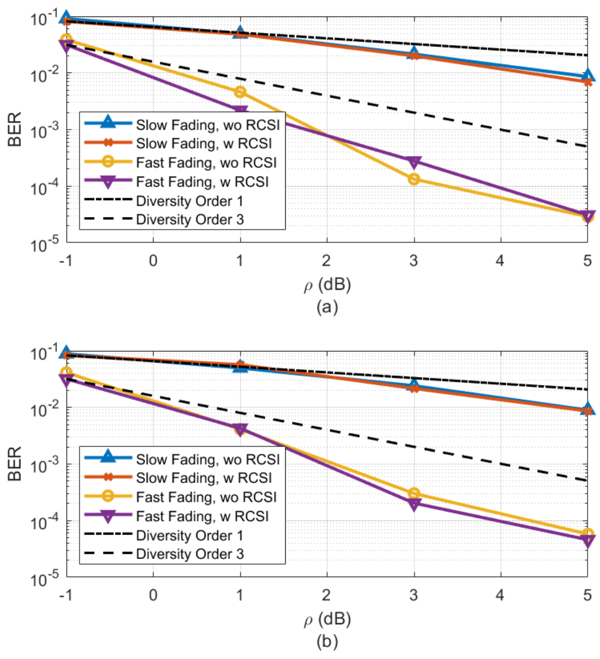


Fig. 6 – The BER curves for DRF codes over Rayleigh magnitude fading channels with $\Omega = 2\sigma^2 = 1$ as a function of the average channel SNR, ρ (dB). (a) Noiseless feedback ($\eta = \infty$) , (b) Noisy feedback ($\eta = 20$ dB)

typically correlated in practice, governed by the Doppler profile of the channel.

In Fig. 6, we compare the resulting BER curves for DRF encoding/decoding over Rayleigh magnitude fading channels when the CSI is and is not available at the receiver (i.e. with and without RCSI). In the case with RCSI, the decoder first performs the Linear Minimum Mean Square Error (LMMSE) channel compensation on the received symbols, i.e., $\hat{y}_i = \frac{\alpha_i}{|\alpha_i|^2+\sigma_n^2}$, and then uses $\hat{y}_i$ as input to the bidirectional LSTM units for decoding. Note that the encoder is the same as depicted in Fig. 2 for both cases. Fig. 6a exhibits the BER curves for the noiseless feedback case ($\eta = \infty$) , and Fig. 6b for the noisy feedback case at $\eta = 20$ dB. For a fair comparison, we use the exact value of the α_i (not an estimated version) both at the encoder and decoder. The curves show similar performance for the two cases with and without CSI at the decoder for both the slow and fast fading cases. Therefore, we can conclude that the proposed DRF code learns to efficiently exploit the knowledge of the instantaneous CSI value α_i available to the encoder through feedback, and no further improvement is achieved by providing the CSI also to the decoder. In other words, regardless of the fading scenario (i.e., slow or fast fading), providing the decoder with the knowledge of perfect instantaneous CSI does not achieve any further improvement in the error rate. This is a desirable result as it shows that using the proposed DRF codes, the complexity and overhead associated with channel estimation at the decoder can be reduced. Finally, note that the dotted curves in Fig. 6 represent the tangent lines with slopes corresponding to diversity orders 1 and 3 for comparison. The DRF codes achieve considerably higher diversity orders compared with the tangent lines specifically in the fast fading case.

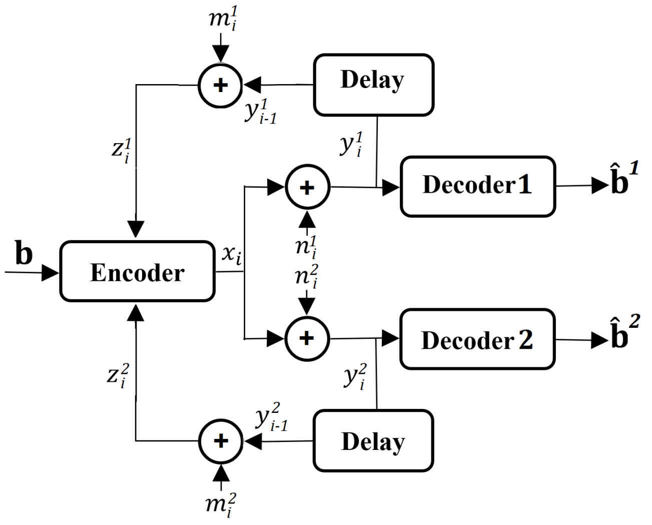


Fig. 7 – The two-user AWGN multicast channel with noisy output feedback and common message

6. MULTICAST CHANNELS WITH FEEDBACK

In a multicast channel with feedback the goal is to transmit a common message to multiple receivers simultaneously while exploiting separate feedback signals from the decoders. This is another example of channels for which we do not have efficient codes with theoretical guarantees. It has been shown that linear feedback approaches that achieve the capacity and improve the error exponent in the case of a single receiver, are strictly suboptimal in the case of multicasting [30]. Linear feedback approaches, even with perfect feedback, fail to achieve the capacity. In the extreme case where the number of receivers goes to infinity, the largest rate achieved by linear feedback schemes tends to be zero [30]. It is clear that non-linear coding schemes are necessary for this channel; however, designing such codes is meticulously difficult. In this section, we show that DRF codes can be employed for multicasting a common message to multiple receivers with a noisy output feedback from each receiver.

Fig. 7 shows the channel model for a multicast AWGN channel with feedback, where the encoder transmits the common message $\mathbf{b} \in \{0,1\}^K$, where K is the message length, to two receivers simultaneously. In this model, x_i denotes the channel input and y_i^1 and y_i^2 are the channel output symbols at receivers 1 and 2, respectively, where $y_i^1 = x_i + n_i^1$ and $y_i^2 = x_i + n_i^2$, and n_i^1 and n_i^2 are jointly Gaussian noise terms, with variances $\sigma_{n^1}^2$ and $\sigma_{n^2}^2$ and correlation coefficient ϵ. The two channel outputs are assumed to be available at the encoder with a unit time delay via passive AWGN feedback channels. At time i, the encoder has a noisy view of what was received by both receivers: $z_i^1 = y_{i-1}^1 + m_i^1$ and $z_i^2 = y_{i-1}^2 + m_i^2$, where m_i^1 and m_i^2 are i.i.d. Gaussian noise terms, i.e., $m_{i-1}^1 \sim \mathcal{N}(0, \sigma_{m^1}^2)$ and $m_{i-1}^2 \sim \mathcal{N}(0, \sigma_{m^2}^2)$. The encoder can use the feedback symbols z_i^1, z_i^2 from the two receivers to sequentially and adaptively decide what to transmit as the next symbol.

Table 3 – BLER values for the two-user AWGN multicast channel with noiseless output feedback ($\eta_1 = \eta_2 = \infty$)

SNR Pair (ρ_1, ρ_2)	$(0, 0)$ dB	$(0, 2)$ dB	$(2, 2)$ dB	$(2, 0)$ dB
$\epsilon = 0$	$(3.3 \times 10^{-1}, 3.2 \times 10^{-1})$	$(3.2 \times 10^{-2}, 1.5 \times 10^{-2})$	$(1.6 \times 10^{-4}, 2.2 \times 10^{-4})$	$(5.3 \times 10^{-3}, 1.2 \times 10^{-2})$
$\epsilon = 0.9$	$(4.7 \times 10^{-3}, 3.8 \times 10^{-3})$	$(1.2 \times 10^{-3}, 3.4 \times 10^{-4})$	$(9.0 \times 10^{-6}, 2.4 \times 10^{-5})$	$(2.5 \times 10^{-4}, 1.0 \times 10^{-3})$
$\epsilon = -0.9$	$(1.3 \times 10^{-1}, 1.3 \times 10^{-1})$	$(6.3 \times 10^{-3}, 1.9 \times 10^{-3})$	$(4.5 \times 10^{-5}, 4.1 \times 10^{-5})$	$(5.6 \times 10^{-3}, 1.0 \times 10^{-2})$
Point to point bound	$(2.0 \times 10^{-4}, 2.0 \times 10^{-4})$	$(2.0 \times 10^{-4}, 5.1 \times 10^{-8})$	$(5.1 \times 10^{-8}, 5.1 \times 10^{-8})$	$(5.1 \times 10^{-8}, 2.0 \times 10^{-4})$

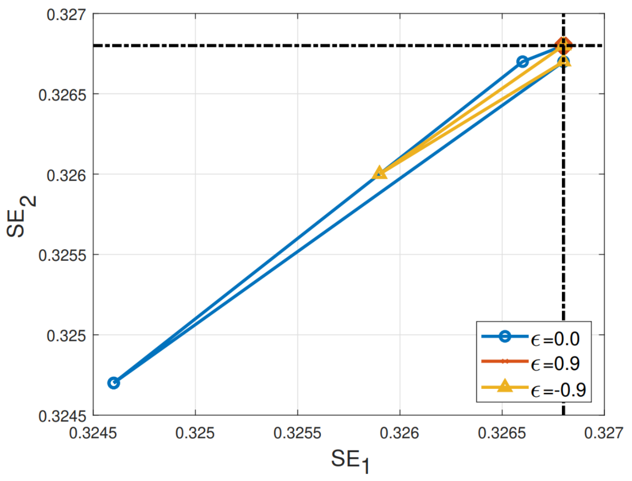


Fig. 8 – The two-user spectral efficiency for a rate $r = 50/153$ DRF code over Guassian multicast channels with feedback ($\eta_1 = \eta_2 = \infty$) .

The encoder maps the message $\mathbf{b} \in \{0,1\}^K$ onto the codeword $\mathbf{x} = [x_1, \ldots, x_L]^T$, where L is the block length and K is the message length. The two decoders map their received codewords $[y_1^1, \ldots, y_L^1]^T$ and $[y_1^2, \ldots, y_L^2]^T$ into the estimated information bits $\hat{\mathbf{b}}^1 \in \{0,1\}^K$ and $\hat{\mathbf{b}}^2 \in \{0,1\}^K$, where $r = K/L$ is the rate of the code. The block error probabilities for the two receivers are given by $Pr\{\hat{\mathbf{b}}^1 \neq \mathbf{b}\}$ and $Pr\{\hat{\mathbf{b}}^2 \neq \mathbf{b}\}$, respectively. As before, we impose an average power constraint on the channel input, i.e., $\frac{1}{L}\mathbb{E}[\|\mathbf{x}\|^2] \leq 1$, where the expectation is over the randomness in the information bits, the randomness in the noisy feedback symbols and any other randomness in the encoder. We denote the forward and feedback channel SNR values for receiver t by $\rho^t = 1/\sigma^2_{n^t}$, and $\eta^t = 1/\sigma^2_{m^t}$, $t = 1, 2$.

The proposed DRF codes provide powerful tools for designing efficient codes for such channels even in the case of noisy feedback. This is achieved by a slight modification of the encoder network to enable the encoder to receive as input the feedback symbols from multiple receivers. The LSTM units determine the parity symbols based on the received feedback from both receivers. This is a challenging task depending on how correlated the two forward noise terms, n_{1i} and n_{2i}, are. The decoder is a two-layer bidirectional LSTM architecture as in the point-to-point case. The loss function is the summation of the binary cross-entropy losses at the two decoders. In general, a weighted sum can be considered to give priority to one of the receivers over the other. We later show through simulations the power of DRF codes in exploiting multiple feedback signals to improve the reliability at both receivers.

In Table 3, we report the BLER pairs achieved for the forward SNR pairs of $(\rho_1, \rho_2) = (0,0), (2,0), (0,2), (2,2)$ dB when the correlation coefficient between the two forward noise sequences is $\epsilon = \{0, 0.9, -0.9\}$. Here, the code rate is $r = 50/153$ and the feedback from both receivers is noiseless. We also provide the BLER values for the point-to-point (single user) case for reference. The two-user BLER values considerably degrade in comparison with the point-to-point case. As expected, the BLER degradation is most when the two forward channel noise sequences are independent. This is due to the fact that when generating parity symbols, the LSTM cells will have to compromise between correcting errors for the two receivers. When these errors are independent, this compromise becomes the most challenging.

For better interpretation, we present the corresponding spectral efficiency values in Fig. 8. The spectral efficiency for receiver r is calculated as $\text{SE}_t = \frac{K \times (1 - \text{BLER}_t)}{L}$, where K and L represent the number of transmitted bits, and the corresponding number of channel uses, respectively, whereas BLER_t is the block error rate at the output of decoder t ($t = 1, 2$) . The dotted black lines in Fig. 8 represent the asymptotic spectral efficiency for each receiver, i.e. $50/153 = 0.3268$. This figure shows that, when the two noise variables n^1 and n^2 are positively correlated, we can achieve performance very close to the error-free spectral efficiency for both users. We lose some performance when the two noises are negatively correlated and the biggest performance loss occurs when they are independent. As expected, encoding becomes the most challenging when the noise variables are independent. Note that the DRF codes can be similarly trained for cases with more than two receivers. Depending on the number of receivers, the training can be resource and time consuming. This is, however, not an issue because training is offline, and once trained, the resulting DRF code is applicable to a wider range of SNR values thanks to our proposed SNR-aware attention mechanism and SNR scheduling for training.

7. CONCLUSIONS

In this paper, we proposed a DNN-based error correction code for fading channels with output feedback, called Deep SNR-Robust Feedback (DRF) code. The proposed encoder transmits the message along with a sequence of parity symbols, which are generated by an LSTM architecture based on the message, as well as the observations of the past forward channel outputs available to the encoder with some additional noise. The decoder is implemented as a two-layer bidirectional LSTM architecture complemented with an SNR-aware attention mechanism. It is shown that the DRF code significantly improves over the existing DNN-based codes in terms of the error rate, as well as the robustness to varying SNR values for AWGN channels with very noisy output feedback. Over fading channels, we showed that DRF codes can learn to efficiently use the knowledge of the instantaneous channel fading (available to the encoder through feedback) to reduce the overhead and complexity associated with channel estimation at the receiver. Finally, we generalized DRF codes to multicast channels with feedback, in which linear feedback codes are known to fall short of achieving the capacity. We showed that DRF codes can improve the reliability of both receivers simultaneously. DRF codes can be extended to many other types of channels, e.g., interference channels or relay channels with feedback, which we leave for future research.

REFERENCES

[1] J. Schalkwijk and T. Kailath. "A coding scheme for additive noise channels with feedback–part I: No bandwidth constraint". In: *IEEE Transactions on Information Theory* 12.2 (1966), pp. 172–182.

[2] J. Schalkwijk. "A coding scheme for additive noise channels with feedback–part II: Band-limited signals". In: *IEEE Transactions on Information Theory* 12.2 (1966), pp. 183–189.

[3] R. G. Gallager and B. Nakiboglu. "Variations on a Theme by Schalkwijk and Kailath". In: *IEEE Transactions on Information Theory* 56.1 (2010), pp. 6–17.

[4] Yury Polyanskiy, H. Vincent Poor, and Sergio Verdu. "Feedback in the Non-Asymptotic Regime". In: *IEEE Transactions on Information Theory* 57.8 (2011), pp. 4903–4925.

[5] C. Shannon. "The zero error capacity of a noisy channel". In: *IRE Transactions on Information Theory* 2.3 (1956), pp. 8–19.

[6] M. Horstein. "Sequential transmission using noiseless feedback". In: *IEEE Transactions on Information Theory* 9.3 (1963), pp. 136–143.

[7] J.M. Ooi and G.W. Wornell. "Fast iterative coding techniques for feedback channels". In: *IEEE Transactions on Information Theory* 44.7 (1998), pp. 2960–2976.

[8] Z. Chance and D. J. Love. "Concatenated Coding for the AWGN Channel With Noisy Feedback". In: *IEEE Transactions on Information Theory* 57.10 (2011), pp. 6633–6649.

[9] Ziad Ahmad, Zachary Chance, David J. Love, and Chih-Chun Wang. "Concatenated Coding Using Linear Schemes for Gaussian Broadcast Channels With Noisy Channel Output Feedback". In: *IEEE Transactions on Communications* 63.11 (2015), pp. 4576–4590.

[10] Kasra Vakilinia, Sudarsan V. S. Ranganathan, Dariush Divsalar, and Richard D. Wesel. "Optimizing Transmission Lengths for Limited Feedback With Nonbinary LDPC Examples". In: *IEEE Transactions on Communications* 64.6 (2016), pp. 2245–2257.

[11] Y. Kim, A. Lapidoth, and T. Weissman. "The Gaussian Channel with Noisy Feedback". In: *2007 IEEE International Symposium on Information Theory*. 2007, pp. 1416–1420.

[12] P. Elias. "Coding for noisy channels". In: *IRE Convention record* 4 (1955), pp. 37–46.

[13] Assaf Ben-Yishai and Ofer Shayevitz. "The Gaussian channel with noisy feedback: Improving reliability via interaction". In: *2015 IEEE International Symposium on Information Theory (ISIT)*. 2015, pp. 2500–2504.

[14] Assaf Ben-Yishai and Ofer Shayevitz. "Interactive Schemes for the AWGN Channel with Noisy Feedback". In: *IEEE Transactions on Information Theory* 63.4 (2017), pp. 2409–2427.

[15] H. Kim, Y. Jiang, S. Kannan, S. Oh, and P. Viswanath. "Deepcode: Feedback Codes via Deep Learning". In: *IEEE Journal on Selected Areas in Information Theory* 1.1 (2020), pp. 194–206.

[16] Yihan Jiang, Hyeji Kim, Himanshu Asnani, Sewoong Oh, Sreeram Kannan, and Pramod Viswanath. "Feedback Turbo Autoencoder". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 8559–8563.

[17] Anahid Robert Safavi, Alberto G. Perotti, Branislav M. Popovic, Mahdi Boloursaz Mashhadi, and Deniz Gündüz. "Deep Extended Feedback Codes". In: *ITU Journal on Future and Evolving Technologies (ITU J-FET)* 2.6 (2021), pp. 33–41.

[18] Yulin Shao, Emre Ozfatura, Alberto Perotti, Branislav Popovic, and Deniz Gündüz. "AttentionCode: Ultra-Reliable Feedback Codes for Short-Packet Communications". In: *IEEE Transactions on Communications* (2023), pp. 1–1. DOI: `10.1109/TCOMM.2023.3280563`.

[19] Emre Ozfatura, Yulin Shao, Alberto G. Perotti, Branislav M. Popović, and Deniz Gündüz. "All You Need Is Feedback: Communication With Block Attention Feedback Codes". In: *IEEE Journal on Selected Areas in Information Theory* 3.3 (2022), pp. 587–602. DOI:`10.1109/JSAIT.2022.3223901`.

[20] Emre Ozfatura, Yulin Shao, Amin Ghazanfari, Alberto Perotti, Branislav Popovic, and Deniz Gündüz. "Feedback is Good, Active Feedback is Better: Block Attention Active Feedback Codes". In: *arXiv preprint arXiv:2211.01730* (2022).

[21] Karl Chahine, Rajesh Mishra, and Hyeji Kim. "Inventing Codes for Channels with Active Feedback via Deep Learning". In: *IEEE Journal on Selected Areas in Information Theory* (2022), pp. 1–1.

[22] Mahdi Boloursaz Mashhadi, Deniz Gündüz, Alberto Perotti, and Branislav Popovic. "DRF Codes: Deep SNR-Robust Feedback Codes". In: *arXiv preprint arXiv:2112.11789* (Dec 2021).

[23] T.A. Summers and S.G. Wilson. "SNR mismatch and online estimation in turbo decoding". In: *IEEE Transactions on Communications* 46.4 (1998), pp. 421–423.

[24] M.A. Khalighi. "Effect of mismatched SNR on the performance of log-MAP turbo detector". In: *IEEE Transactions on Vehicular Technology* 52.5 (2003), pp. 1386–1397.

[25] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". In: *arXiv:1409.0473v7* (May 2016).

[26] Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. "Massive Exploration of Neural Machine Translation Architectures". In: *arXiv:1703.03906v2* (Mar 2017).

[27] Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. "Structured attention networks". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. 2017.

[28] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. "A decomposable attention model". In: *Empirical Methods in Natural Language Processing*. 2016.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. "Attention Is All You Need". In: *Neural Information Processing Systems*. 2017.

[30] Youlong Wu, Paolo Minero, and Michèle Wigger. "Insufficiency of Linear-Feedback Schemes in Gaussian Broadcast Channels With Common Message". In: *IEEE Transactions on Information Theory* 60.8 (2014), pp. 4553–4566.

[31] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber. "LSTM: A Search Space Odyssey". In: *IEEE Transactions on Neural Networks and Learning Systems* 28.10 (2017), pp. 2222–2232.

[32] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. "On the difficulty of training recurrent neural networks". In: *Proceedings of the 30th International Conference on Machine Learning*. 2013, pp. 1310–1318.

[33] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. "An empirical exploration of recurrent network architectures". In: *Proceedings of the 32th International Conference on Machine Learning*. 2015, pp. 2342–2350.

[34] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[35] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. "Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour". In: *ArXiv* 1706.02677v2 (2018).

[36] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. "Curriculum learning". In: *26th Annual International Conference on Machine Learning (ICML 2009)*. June 2009, pp. 41–48.

[37] Alex Graves, Marc G. Bellemare, Jacob Menick, Rémi Munos, and Koray Kavukcuoglu. "Automated Curriculum Learning for Neural Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. 2017, pp. 1311–1320.

[38] "3GPP Technical Specification 38.214 - NR; Physical layer procedures for data (Release 16)". In: ().

[39] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[40] Huawei-HiSilicon. "Performance evaluation of LDPC codes for NR eMBB data". In: *R1-1713740, 3GPP RAN1 meeting 90, Prague, Czech Republic* (August 21–25, 2017).

[41] Ezio Biglieri, John Proakis, and Shlomo Shamai. "Fading channels: Information-theoretic and communications aspects". In: 44.6 (1998), pp. 2619–2692.

[42] Marvin K Simon and Mohamed-Slim Alouini. "A unified approach to the performance analysis of digital communication over generalized fading channels". In: 86.9 (1998), pp. 1860–1877.

AUTHORS

Mahdi Boloursaz Mashhadi (Senior Member, IEEE) is a lecturer at the 5G/6G Innovation Centre (5G/6GIC) at the Institute for Communication Systems (ICS), University of Surrey (UoS). Prior to joining ICS, he was a postdoctoral research associate at the Intelligent Systems and Networks (ISN) Research Group, Imperial College London, 2019-2021. He received B.S., M.S., and Ph.D. degrees in mobile telecommunications from the Sharif University of Technology (SUT), Tehran, Iran, in 2011, 2013, and 2018, respectively. His research interests include wireless communications, signal processing, AI and machine learning. He has served as a panel judge for the International Telecommunication Union (ITU) to evaluate innovative submissions on applications of AI/ML in 5G and beyond wireless networks since 2021.

Deniz Gündüz (Fellow, IEEE) received B.S. degree in electrical and electronics engineering from the Middle East Technical University, Turkey, in 2002, and M.S. and Ph.D. degrees in electrical engineering from the NYU Tandon School of Engineering (formerly Polytechnic University) in 2004 and 2007, respectively. He is currently a professor of information Processing with the Electrical and Electronic Engineering Department, Imperial College London, U.K. His research interests lie in the areas of communications and information theory, machine learning, and privacy. He is a recipient of the Consolidator (2022) and Starting (2016) Grants of the European Research Council, the IEEE Communications Society-Communication Theory Technical Committee Early Achievement Award in 2017, and several best paper awards. He is an area editor for the IEEE Transactions on information theory, IEEE Transactions on communications, IEEE Journal on selected areas in communications, and Special Series on Machine Learning in Communications and Networks.

Alberto G. Perotti (Senior Member, IEEE) received a Ph.D. degree in telecommunications from the Politecnico di Torino, Italy, in 2003. He is a principal research engineer with Huawei Technologies, where he is involved in wireless networks' physical layer research and standardization. Prior to joining Huawei, he held positions with the Politecnico di Torino and has been the head of Networks and Wireless Communications Research with CSP-ICT Innovation, Turin, Italy. His research covers channel coding and modulation, multiple access, applications of machine learning to wireless, and software defined radios.

Branislav M. Popovic received a Ph.D. degree in electrical engineering from the School of Electrical Engineering, University of Belgrade, Serbia. Prior to joining Huawei Technologies, Stockholm, in 2001, he was with the Institute of Microwave Techniques and Electronics, Belgrade, from 1984 to 1994, Ericsson, Stockholm, from 1994 to 2000, and Marconi, Stockholm, from 2000 to 2001. He is a Huawei Fellow.