Learning-Based Near-Orthogonal Superposition Code for MIMO Short Message Transmission

Chenghong Bian, Chin-Wei Hsu, Changwoo Lee, and Hun-Seok Kim, *Senior Member, IEEE*

***Abstract*— Massive machine type communication (mMTC) has attracted new coding schemes optimized for reliable short message transmission. In this paper, a novel deep learning-based near-orthogonal superposition (NOS) coding scheme is proposed to transmit short messages in multiple-input multiple-output (MIMO) channels for mMTC applications. In the proposed MIMO-NOS scheme, a neural network-based encoder is optimized via end-to-end learning with a corresponding neural network-based detector/decoder in a superposition-based auto-encoder framework including a MIMO channel. The proposed MIMO-NOS encoder spreads the information bits to multiple near-orthogonal high dimensional vectors to be combined (superimposed) into a single vector and reshaped for the space-time transmission. For the receiver, we propose a novel looped K-best tree-search algorithm with cyclic redundancy check (CRC) assistance to enhance the error correcting ability in the block-fading MIMO channel. For a comprehensive understanding of the proposed MIMO-NOS scheme, we further quantify the gain from individual components/modules in the framework, and analyze the decoding complexity measured by the floating point operations (FLOPs). Simulation results show the proposed MIMO-NOS scheme outperforms maximum likelihood (ML) MIMO detection combined with a polar code with CRC-assisted list decoding by 1 – 2 dB in various MIMO systems for short (32 – 64 bit) message transmission.**

***Index Terms*— Massive machine type communications, near-orthogonal modulation, superposition coding, learned modulation, learned coding, MIMO.**

I. Introduction

MASSIVE machine type communication (mMTC) is an essential technology for next generation wireless standards to enable a wide range of applications including health, security and transportation [2], [3], [4]. These applications, by nature, typically employ short messages/packets carrying a relatively small number of information bits, which make conventional codes designed with a long block length assumption less effective with relatively small error exponents and/or non-negligible coding gain losses. Polar codes with list decoding [5] are proven to be more reliable compared with other modern codes such as LDPC and turbo codes for short block lengths [6]. However, their performance is far from capacity, and thus new coding schemes have been actively investigated for short message transmission [7].

Hyper-dimensional modulation (HDM) is a recently proposed non-orthogonal modulation scheme for short packet communications [8], [9], [10]. HDM can be seen as a joint coding-modulation method and a special type of superposition codes [11]. Instead of using a random codebook as in typical superposition codes, HDM uses Fast Fourier Transform (FFT) and pseudo-random permutations to encode sparse pulse position modulated information vectors to a non-sparse superimposed hyper-dimensional vector for efficient encoding and decoding. HDM was first proposed with a demodulation algorithm using an iterative parallel successive interference cancellation (SIC) technique [8]. It is then extended using a K-best decoding algorithm [10] in AWGN and interference-limited channels to outperform the state-of-the-art CRC-assisted polar codes [5] applied to binary phase-shift keying (BPSK) under the same spectral efficiency. Despite its excellent reliability and low complexity for short message packets, the hand-crafted encoding scheme using FFT and pseudo-random permutation for codeword generation in HDM is sub-optimal. It is shown in our prior work [1] that a deep learning-based near-orthogonal superposition (NOS) encoding scheme can outperform HDM in single antenna additive white Gaussian noise (AWGN) channels.

In order to overcome the limitations of hand-crafted coding and modulation schemes, data-driven learning with deep neural networks (DNNs) has been applied to the realm of channel coding and modulation [12], [13], [14], [15], [16]. One of early applications of DNNs is to decode linear block codes replacing hand-crafted decoding algorithms for polar and LDPC codes with unmodified encoders. Taking advantage of powerful deep learning, prior schemes [12] and [13] show improved decoding performance and enhanced robustness under various channel conditions. Meanwhile, new channel codes have been recently investigated via end-to-end learning. A DNN-based learned code was originally introduced in [14] where the encoder learns a joint coding and modulation scheme generating a length-7 codeword from a length-16 one-hot input to achieve the performance similar to that of (7, 4) Hamming code. The authors in [15] propose an RNN-based auto-encoder that emulates a convolutional code (CC) which takes the bit-sequence input instead of processing an one-hot encoded input vector.

Manuscript received 30 June 2022; revised 4 December 2022 and 13 March 2023; accepted 28 April 2023. Date of publication 9 May 2023; date of current version 18 September 2023. This work was funded in part by NSF CAREER #1942806. An earlier version of this paper was presented in part at the IEEE International Conference on Communications (ICC), 2022 [DOI: 10.1109/ICC45855.2022.9838685]. The associate editor coordinating the review of this article and approving it for publication was M. Flanagan. *(Corresponding author: Chenghong Bian.)*

Chenghong Bian is with the Department of Electrical and Electronic Engineering, Imperial College London, SW7 2AZ London, U.K. (e-mail: c.bian22@imperial.ac.uk).

Chin-Wei Hsu, Changwoo Lee, and Hun-Seok Kim are with the Department of Electrical and Computer Engineering, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: chinweih@umich.edu; cwoolee@umich.edu; hunseok@umich.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2023.3274158.

Digital Object Identifier 10.1109/TCOMM.2023.3274158

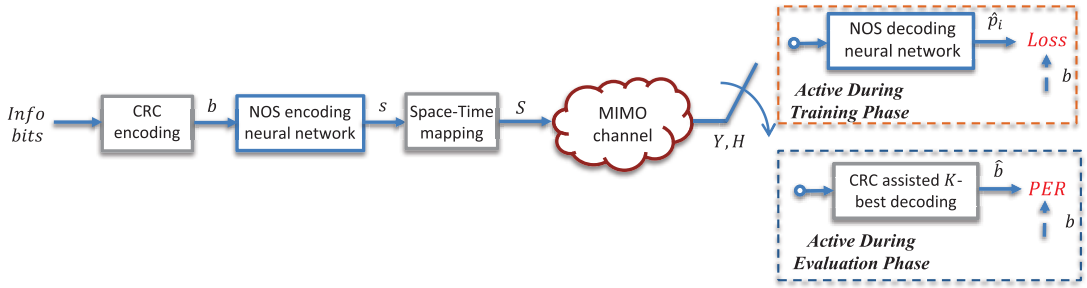


Fig. 1. Encoding and decoding flow of the proposed MIMO-NOS scheme.

This *learned* CC outperforms conventional CC to attain lower bit/packet error rates (BER/PER). In [16], the authors propose a learned turbo auto-encoder which employs convolutional neural networks (CNNs) and interleaving. The decoder in [16] unfolds the iterative decoding process to multiple DNN layers to achieve the BER performance that is comparable to that of the conventional turbo code.

Meanwhile, researchers are actively extending deep learning to MIMO detection problems. DetNet proposed in [17] unfolds the projected gradient descent algorithm via deep learning to achieve near optimal detection performance with significantly improved running speed. The authors in [18] further extend the topic to joint MIMO detection and polar decoding, where a DNN-based receiver takes both the received signal and the estimated channel state information (CSI) as the input to produce the estimated information sequence output. Their evaluation shows the DNN-based joint detection and decoding scheme outperforms the conventional iterative MIMO receiver where soft-decision information is exchanged between a sphere decoder and a polar decoder to achieve near-optimal performance. However, their DNN-based receiver can only handle very short packets with 16 information bits and each MIMO channel configuration requires a specifically trained neural network model. These limitations make it rather impractical for emerging mMTC applications.

Inspired by aforementioned HDM and deep learning-based coding, we originally introduced a DNN-based near-orthogonal superposition (NOS) coding scheme in [1] to learn a near-orthogonal codebook for superimposed transmission of short packets in single-input single-output AWGN channels. In this paper, we further extend the NOS code to MIMO configurations constructing a *learned MIMO-NOS coding scheme*. In our approach, an information bit sequence is first appended by cyclic redundancy check (CRC) bits to improve the reliability in low signal-to-noise ratio (SNR) scenarios. The CRC appended bit sequence $\boldsymbol{b}$ is transformed into several one-hot coded vectors which are fed into the MIMO-NOS encoder followed by a simple space-time coding (STC) block that maps the encoder output to different transmit antennas and time slots. To learn a good MIMO-NOS codebook, a DNN-based receiver that integrates a residual-assisted minimum mean square error (MMSE) MIMO equalizer/detector along with a neural decoder is jointly trained to enable end-to-end back-propagation through the encoder, MIMO channel model, and decoder. Upon learning a good MIMO-NOS codebook, we employ a CRC-assisted looped K-best tree-search decoding algorithm to improve the error rate performance beyond the limitation of the learned MIMO detector/decoder used for the training. The overall datapath of the proposed MIMO-NOS scheme is shown in Fig.1.

The main contributions of this paper are summarized as follows:

1) A novel deep learning-based near-orthogonal superposition code is proposed for reliable short packet transmission in MIMO channels. We combine individual modules including channel coding, modulation, MIMO detection, and channel decoding into a single deep learning model and optimize it via end-to-end training. As a result, the MIMO-NOS encoder learns a set of superposition codes with desired properties in the high dimensional codeword space. Moreover, to the best of our knowledge, it is the first work that jointly learns these individual modules for MIMO communications in an end-to-end fashion.
2) A new CRC-assisted looped K-best decoding algorithm is designed to outperform the DNN-based receiver used during the training. The proposed decoding algorithm finds the top-K bit-sequences maximizing the (approximated) posterior probability to significantly improve the PER performance beyond the capability of the DNN-based receiver utilized to learn the MIMO-NOS codebook.
3) Numerical characterization of the learned MIMO-NOS codebook is provided to understand the codebook properties, derive detection/decoding metrics for the looped K-best decoder, and study the performance of the proposed algorithm. It is also shown that the learned MIMO-NOS codebook can be applied to different MIMO configurations with robust performance via simple space-time mapping without retraining the encoder network.
4) Extensive numerical evaluations are performed to quantify the gain of the proposed learned MIMO-NOS scheme compared to the ML MIMO detection with CRC-aided list decoding polar codes, which is one of the state-of-the-art baseline schemes.
5) Detailed analysis is provided for the decoding complexity measured by the number of FLOPs as well as the decoding latency. The proposed looped K-best decoder can achieve lower decoding latency with improved PER performance compared to the baseline that uses a polar code with ML MIMO detection.

6) Extensive simulations to quantify the gain from individual components of the proposed scheme such as the learned coded modulation, space-time mapping at the transmitter, and the joint detection and decoding at the receiver. These experiments provide more insights leading to a comprehensive understanding of the proposed scheme.

Throughout the paper, scalar variables are represented with normal-face letters x while matrices and vectors with **upper** and **lower** case letters, $\boldsymbol{X}$ and $\boldsymbol{x}$, respectively. A set is denoted by $\mathbb{S}$. Transpose and Hermitian operators are denoted by $(\cdot)^{\top}$, $(\cdot)^{\dagger}$, respectively. $\Re(x)$ ($\Im(x)$) denotes the real (imaginary) part of a complex variable x. Moreover, $vec(\boldsymbol{X})$ transforms the matrix $\boldsymbol{X}$ into a column vector $\boldsymbol{x}$ by stacking the columns of $\boldsymbol{X}$. Finally, we denote the Frobenius norm of matrix $\boldsymbol{X}$ as $||\boldsymbol{X}||_F$.

II. MIMO-NOS Code Learning

In this section, we briefly recap the conventional coded MIMO transceiver for the baseline, and then introduce the neural network structure for the proposed MIMO-NOS scheme and its training methodology.

A. Conventional MIMO Transceiver

Let $\boldsymbol{b}$ and $\boldsymbol{c}$ denote the information bit sequence and the corresponding coded bit sequence with length $N_t M_c \log_2 Q$ bits. The coded sequence $\boldsymbol{c}$ is interleaved and then mapped to a matrix $\boldsymbol{S}$ of dimension $N_t \times M_c$ whose entries are chosen from a complex constellation set (e.g., QPSK) with Q symbols. N_t is the number of transmit antennas and M_c is the number of MIMO channel uses for transmission. The received signal $\boldsymbol{Y} \in \mathbb{C}^{N_r \times M_c}$ with N_r receive antennas can be written as:

$$\boldsymbol{Y} = \boldsymbol{H}\boldsymbol{S} + \boldsymbol{N}, \tag{1}$$

where $\boldsymbol{H} \in \mathbb{C}^{N_r \times N_t}$ is the complex MIMO channel which is assumed to be perfectly known to the receiver and $\boldsymbol{N}$ is the complex Gaussian noise whose entries are i.i.d. with zero mean and element-wise variance $N_t\sigma^2$. Note that the N_t in the noise variance unifies the SNR definition for MIMO systems adopting different numbers of transmit antennas as discussed later. In this paper, we assume each element of $\boldsymbol{H}$ is an i.i.d. complex Gaussian random variable with zero mean and unit variance. $\boldsymbol{H}$ is randomly realized for each and every packet.

There are numerous MIMO detection algorithms to solve (1) and obtain soft decisions of bits in the matrix $\boldsymbol{S}$ with a simplifying assumption that bits in $\boldsymbol{c}$ are independent. With that assumption, the ML MIMO detector is the optimal scheme,[1] and thus it is applied to the baseline as briefly introduced in the following.

Consider the log-likelihood ratio (LLR) L for a certain bit c_k from $\boldsymbol{c}$ given $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{s} + \boldsymbol{n}$ where $\boldsymbol{s}$ is a $N_t \times 1$ transmit vector (a column of $\boldsymbol{S}$ that involves c_k), while $\boldsymbol{y}$ and $\boldsymbol{n}$ are the corresponding received and noise vectors, respectively. The LLR of c_k can be obtained by

$$L(c_k|\boldsymbol{y}, \boldsymbol{H}) = \ln \frac{P[c_k = +1|\boldsymbol{y}, \boldsymbol{H}]}{P[c_k = -1|\boldsymbol{y}, \boldsymbol{H}]}. \tag{2}$$

Applying Bayes' rule and assuming equal probability of the bit symbols, the L values are obtained by

$$L(c_k|\boldsymbol{y}, \boldsymbol{H}) = \ln \frac{\sum_{\boldsymbol{x} \in \mathbb{X}_{k,+1}} \exp\{\frac{-||\boldsymbol{y}-\boldsymbol{H}\boldsymbol{x}||_2^2}{2N_t\sigma^2}\}}{\sum_{\boldsymbol{x} \in \mathbb{X}_{k,-1}} \exp\{\frac{-||\boldsymbol{y}-\boldsymbol{H}\boldsymbol{x}||_2^2}{2N_t\sigma^2}\}} \tag{3}$$

where each set $\mathbb{X}_{k,+1} = \{\boldsymbol{x}|c_k = +1\}$ or $\mathbb{X}_{k,-1} = \{\boldsymbol{x}|c_k = -1\}$ contains $2^{N_t \log_2 Q - 1}$ bit sequences of length $N_t \log_2 Q$ bits, enumerating all possible bit sequences given $c_k = +1$ or -1. In the baseline scheme, we first calculate the LLR (i.e., soft decision) for each coded bit using (3), deinterleave the LLR sequence, and then feed it into the subsequent soft-input channel decoder (such as a CRC-assisted list polar decoder) to recover the original information bit sequence $\mathbf{b}$.

B. MIMO-NOS Coding

In this subsection, we first briefly recap the NOS code designed for the single antenna AWGN channel and then extend this scheme to MIMO systems. The NOS code belongs to the general class of superposition code whose encoding is described as follows:

Consider a sequence of information bits $\boldsymbol{b}$ whose length is $N_E \times m$ bits, where N_E is the number of encoders in our proposed superposition coding scheme. It is split into N_E smaller bit sequences $\boldsymbol{b}_j, j = 1, \cdots, N_E$ each carrying m bits. Each $\boldsymbol{b}_j$ is converted to an one-hot vector $\boldsymbol{x}_j$ with length $M = 2^m$ whose only non-zero position (with value 1) is determined by $\boldsymbol{b}_j$. A superposition code is defined by a family of complex-valued codebooks $\boldsymbol{\mathcal{C}}_j, j \in [1, N_E]$ each with dimension $(n/2) \times M$ where $n/2$ is the codeword length. The codeword corresponding to the bit sequence $\boldsymbol{b}_j$ is obtained by $\boldsymbol{\mathcal{C}_j}@_j$ (i.e., matrix-vector multiplication), whose dimension is $(n/2) \times 1$. The superimposed transmit vector $\boldsymbol{s}$ with length $n/2$ for the entire bit sequence $\boldsymbol{b}$ is then obtained by adding (superimposing) N_E codewords such that:

$$\boldsymbol{s} = \sum_{j=1}^{N_E} \boldsymbol{\mathcal{C}}_j x_j. \tag{4}$$

Conventional superposition codes adopt pseudo-random codebooks, e.g., random (complex) Gaussian codebooks as in [11], whereas a more efficient scheme such as HDM [8] defines the codebook using the Discrete (Fast) Fourier Transform (DFT/FFT) matrix along with pseudo-random permutations. There exist efficient decoding algorithms for these schemes in the AWGN channel including successive interference cancellation (SIC) [8], [11] and approximate message passing (AMP) [19] algorithms. Although these superposition codes are proven to be capacity achieving when the block length goes to infinity [11], a pseudo-random codebook is shown to be less effective under short block lengths [20]. Thus, we proposed a new NOS code in our prior work [1] where

[1]The complexity of ML detection may be prohibitive for a large MIMO system with a high-order constellation. Low complexity close-to-ML algorithms are available but they are beyond our consideration for this paper.

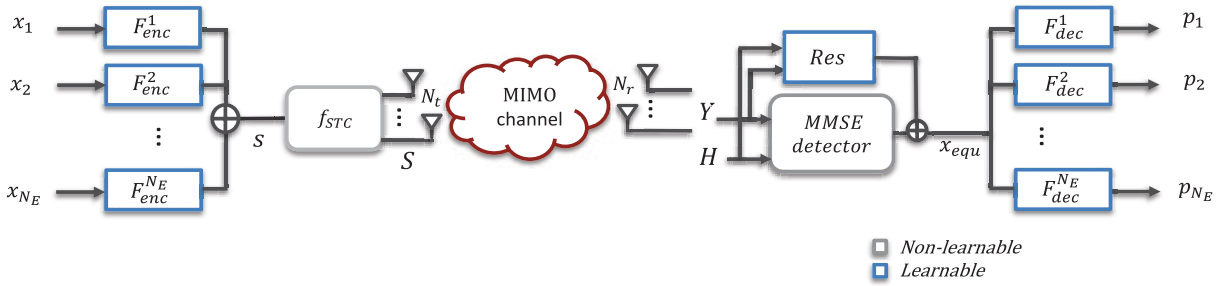


Fig. 2. The proposed neural network-based MIMO-NOS encoding and decoding structure for training.

the codebook is optimized via end-to-end learning with the assistance of a neural network decoder and the resulting codewords belonging to different codebooks $\mathcal{C}_i, \mathcal{C}_j, i \neq j$, are near-orthogonal. At the receiver, the NOS decoder first estimates the information vectors and then performs a cyclic redundancy check (CRC)-assisted K-best tree-search algorithm to reduce the packet/bit error rate. This prior scheme was designed for single antenna cases where the near-orthogonal property of the codewords is maintained after the AWGN channel. However, the same property does not hold when the codewords pass through MIMO channels. In this work, we propose an extended MIMO-NOS scheme for short message MIMO transmissions.

Fig. 2 shows the overview of the proposed MIMO-NOS transmission scheme. Multiple (N_E) one-hot vectors $\boldsymbol{x}_j$ are fed to dedicated neural network-based encoders F^j_{enc} to generate real-valued coded vectors $\tilde{\boldsymbol{s}}_j = F^j_{enc}(x_j)$ of length n. Each $F^j_{enc}, j \in [1, N_E]$ has the same neural network structure that consists of linear layers, batch normalization layers, and non-linear activation functions. Since each $\tilde{\boldsymbol{s}}_j$ conveys the same amount of information, we assign the same energy $||\tilde{\boldsymbol{s}}_j||_2^2 = \frac{n}{2N_E}$ to each $\tilde{\boldsymbol{s}}_j$ using a power normalization layer at the end of each F^j_{enc}. Instead of transmitting a real-valued signal, we convert the length-n real-valued vector $\tilde{\boldsymbol{s}}_j$ into a complex vector $\boldsymbol{s}_j = \tilde{\boldsymbol{s}}_j^R + i\tilde{\boldsymbol{s}}_j^I$ to improve spectral efficiency where $\tilde{\boldsymbol{s}}_j^R$ and $\tilde{\boldsymbol{s}}_j^I$ are $(n/2) \times 1$ vectors obtained by splitting $\tilde{\boldsymbol{s}}_j$ so that $\tilde{\boldsymbol{s}}_j = \begin{bmatrix} \tilde{\boldsymbol{s}}_j^R \\ \tilde{\boldsymbol{s}}_j^I \end{bmatrix}$ holds. The superimposed signal $\boldsymbol{s}$ is obtained by adding all $\boldsymbol{s}_j, j = 1, 2, \ldots, N_E$:

$$\boldsymbol{s} = \sum_{j=1}^{N_E} \boldsymbol{s}_j. \tag{5}$$

Then, we map $\boldsymbol{s}$ to different transmit antennas and time slots for space-time coding, which is extensively studied in [21] and [22]. In this paper, we define a reshape function f_{STC} as a simple space-time coding scheme that converts/reshapes the $(n/2) \times 1$ input vector, $\boldsymbol{s}$, to a matrix $\boldsymbol{S} \in \mathbb{C}^{N_t \times M_c}$ where N_t is the number of transmit antennas, M_c is the number of channel uses (or time slots), and $N_t M_c = n/2$ holds. To be precise, $\boldsymbol{S} = f_{STC}(\boldsymbol{s})$ and $\boldsymbol{S}_{i,j} = \boldsymbol{s}_{i+(j-1)M_c}$ is satisfied.

We assume the block-fading (quasi-static) MIMO channel, where channel coefficients in $\boldsymbol{H} \in \mathbb{C}^{N_r \times N_t}$ are instantiated as i.i.d. complex Gaussian random variables that remain constant for a single block transmission (M_c channel uses). The next block observes an independent random channel realization $\boldsymbol{H}$ following the same model used for the conventional MIMO transmission. The received signal $\boldsymbol{Y}$ after the MIMO channel can be expressed as $\boldsymbol{Y} = \boldsymbol{HS} + \boldsymbol{N}$ as in (1) where $\boldsymbol{N} \in \mathbb{C}^{N_r \times M_c}$ is the complex Gaussian noise whose entries are i.i.d. with zero mean and element-wise variance $N_t\sigma^2$. The signal to noise ratio (SNR) of the system is defined as:

$$SNR = \frac{\mathbb{E}(||\boldsymbol{HS}||_F^2)}{\mathbb{E}(||\boldsymbol{N}||_F^2)} = \frac{N_r n/2}{N_t N_r M_c \sigma^2} = \frac{1}{\sigma^2}, \tag{6}$$

where we use the fact that $N_t M_c = n/2$ and $\mathbb{E}(||\boldsymbol{S}||_F^2) = n/2$ ($\boldsymbol{s}_j$ are near-orthogonal to each other as examined in the later section). Note that, in practical systems, $\boldsymbol{H}$ can be obtained at the receiver by applying channel estimation algorithms [23], [24], [25] to the received pilots. However, we make a simplifying assumption that $\boldsymbol{H}$ is perfectly available at the receiver throughout this paper.

C. Learned MIMO-NOS Receiver

To learn a set of MIMO-NOS encoders $F^j_{enc}, j \in [1, N_E]$, the training process uses a matching set of MIMO-NOS decoders. For decoding, the received signal $\boldsymbol{Y}$ and the MIMO channel $\boldsymbol{H}$ are first fed to the residual-assisted MIMO detector/equalizer that consists of a conventional MMSE equalization module which serves as the backbone and a residual connection neural network module to compensate the output of the MMSE equalization module as shown in Fig. 2. This residual-assisted structure is inspired by [26] and it outperforms the MMSE-only structure as well as the neural network-only structure.

The MMSE equalization module output is:

$$\boldsymbol{X}_{MMSE} = (\boldsymbol{H}^\dagger \boldsymbol{H} + \frac{1}{SNR}\boldsymbol{I}_{N_t})^{-1}\boldsymbol{H}^\dagger \boldsymbol{Y}, \tag{7}$$

where SNR is defined in (6). The residual module, shown in Fig. 3, is a neural network defined as $Res(\cdot)$ that takes the real-valued[2] received signal $\tilde{\boldsymbol{Y}} \in \mathbb{R}^{2N_r \times M_c}$ and the real-valued vectorized CSI, $\tilde{\boldsymbol{h}} \in \mathbb{R}^{2N_t N_r \times 1}$ as input and outputs calibration information for the MMSE equalization module. To be precise, we first duplicate $\tilde{\boldsymbol{h}}$ M_c times in a column-wise manner to form a matrix $\tilde{\boldsymbol{H}} \in \mathbb{R}^{2N_t N_r \times M_c}$, $\tilde{\boldsymbol{H}} = [\tilde{\boldsymbol{h}}, \tilde{\boldsymbol{h}}, \ldots, \tilde{\boldsymbol{h}}]$. We then concatenate each columns of

[2]Note that the real valued signals are obtained by concatenating the real and imaginary parts of their complex counterpart.

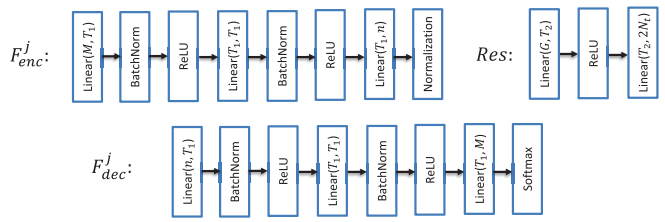


Fig. 3. Structures of each encoder decoder pair F^j_{enc} and $F^j_{dec}, j \in [1, N_E]$ and the residual network Res. T_1 and T_2 denote the number of hidden neurons in the model.

$\tilde{\boldsymbol{Y}}$ and $\tilde{\boldsymbol{H}}$ to generate $\tilde{\boldsymbol{Y}}_{ext} = \begin{bmatrix} \tilde{\boldsymbol{Y}} \\ \tilde{\boldsymbol{H}} \end{bmatrix}$ where $\tilde{\boldsymbol{Y}}_{ext} \in \mathbb{R}^{G \times M_c}$ with $G = 2N_t(N_r + 1)$. Then Res neural network module processes this concatenated signal $\tilde{\boldsymbol{Y}}_{ext}$ to produce the final output of the residual-assisted MIMO detector expressed by:

$$\tilde{\boldsymbol{X}}_{EQU} = \tilde{\boldsymbol{X}}_{MMSE} + Res(\tilde{\boldsymbol{Y}}_{ext}), \tag{8}$$

where $\tilde{\boldsymbol{X}}_{MMSE} \in \mathbb{R}^{2N_t \times M_c}$ is the real-valued version of the MMSE equalization module output.

We then vectorize $\tilde{\boldsymbol{X}}_{EQU}$ to $\tilde{\boldsymbol{x}}_{equ}$ with n-dimension, which is fed into each decoder $F^j_{dec}, j \in [1, N_E]$ as shown in Fig. 2. Similar to F^j_{enc}, these $F^j_{dec}, j \in [1, N_E]$ are neural networks that consist of linear layers, batch normalization layers, and non-linear activation functions. Each F^j_{dec} for a specific j is trained to produce/estimate the probability vector $\boldsymbol{p}_j = F^j_{dec}(\tilde{x}_{equ})$. The length of $\boldsymbol{p}_j$ is M and $\boldsymbol{p}_j[m]$ represents the probability of $x_j[m] = 1$.

The detailed neural network structures of F^j_{enc}, F^j_{dec}, and Res are shown in Fig. 3 where T_1, T_2 denote the number of neurons in the hidden layers. The training of F^j_{enc}, F^j_{dec}, and Res is performed by optimizing the cross-entropy loss for each pair of the one-hot input $\boldsymbol{x}_j$ and the probability vector $\boldsymbol{p}_j$. Since $\boldsymbol{x}_j$'s assigned to different F^j_{enc}'s are independent of each other, the total loss is the summation of pairwise losses:

$$loss = -\sum_{j=1}^{N_E} \sum_{m=1}^{M} \boldsymbol{x}_j[m] \log(\boldsymbol{p}_j[m]). \tag{9}$$

We randomly generate different bit sequences $\boldsymbol{b}$ and MIMO channel realizations $\boldsymbol{H}$ in the training phase, and adopt the ADAM optimizer to minimize the loss in (9) corresponding to different $\boldsymbol{b}, \boldsymbol{H}$ realizations to train the proposed neural networks.

III. The Learned MIMO-NOS Codebook Properties

In this section, we inspect the properties of the learned MIMO-NOS codebook before and after the MIMO channel.

A. Codebook Properties Before the MIMO Channel

Since we divide the input bit sequence into N_E shorter sequences for separate encoding using F^j_{enc} $(j = 1, 2, \ldots, N_E)$, the dimension of learned complex-valued codebook $\boldsymbol{\mathcal{C}}_j$ is significantly smaller compared to that of a conventional linear block code that encodes the entire input bit sequence. Our codebook $\boldsymbol{\mathcal{C}}_j$ has the dimension of $(n/2) \times M$, and it is obtained by enumerating all length-M one-hot vectors for each encoder F^j_{enc} after successful training:

$$\boldsymbol{\mathcal{C}}_j[m] = \tilde{\boldsymbol{s}}^R_{j,m} + i\tilde{\boldsymbol{s}}^I_{j,m}, \tag{10}$$

where $\boldsymbol{\mathcal{C}}_j[m]$ denotes the m-th column of $\boldsymbol{\mathcal{C}}_j$, $\tilde{\boldsymbol{s}}^R_{j,m}$ and $\tilde{\boldsymbol{s}}^I_{j,m}$ are $(n/2) \times 1$ vectors obtained by splitting $F^j_{enc}(x_m)$ so that $F^j_{enc}(x_m) = \begin{bmatrix} \tilde{\boldsymbol{s}}^R_{j,m} \\ \tilde{\boldsymbol{s}}^I_{j,m} \end{bmatrix}$ holds.

Similar to the analysis in [1] for single antenna channels, we first analyze the properties of the constructed codebook by observing the absolute values of inner products between codewords belonging to different encoders. This forms a cross-correlation tensor c_{inter} with dimension $N_E \times (N_E - 1) \times M \times M$ which is defined as:

$$c_{inter}[i, j, k, l] = \frac{|\Re(\boldsymbol{\mathcal{C}}^\dagger_i[k]\boldsymbol{\mathcal{C}}_j[l])|}{n/(2N_E)}$$
$$i, j \in [1, N_E]; i \neq j; k, l \in [1, M]. \tag{11}$$

Note that c_{inter} quantifies the level of interference from the codewords belonging to different encoders, thus it represents the *inter*-correlation property of the codebook. We further evaluate inner products between the codewords belonging to the *same* encoder to form another tensor c_{intra} with dimension $N_E \times M \times (M - 1)$ defined as:

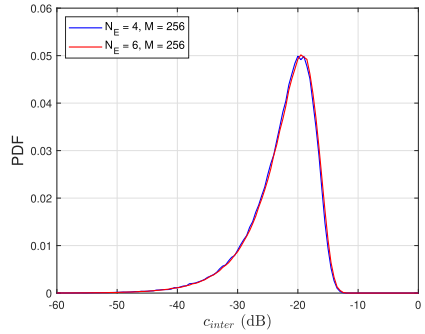
$$c_{intra}[i, k, l] = \frac{\Re(\boldsymbol{\mathcal{C}}^\dagger_i[k]\boldsymbol{\mathcal{C}}_i[l])}{n/(2N_E)}$$
$$i \in [1, N_E]; k \neq l; k, l \in [1, M], \tag{12}$$

which represents the *intra*-correlation property. Since the power of codewords are normalized, c_{intra} directly reflects the L2-distance between codewords belonging to the same encoder. A small (or negative) c_{intra} entry implies longer L2-distance for the corresponding pair, which is desirable to lower the error rate. Since the error performance of a code is mainly determined by its minimum distance, we are interested in the distribution of entries of c_{intra} with relatively large positive values.

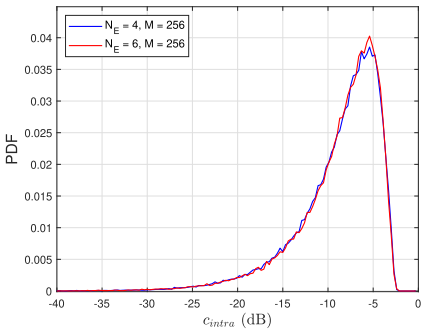
Fig. 4(a) shows the distribution of entries (in dB) in c_{inter} for the codebooks trained with $(N_E = 4, M = 256, n = 64, N_t = N_r = 2)$ and $(N_E = 6, M = 256, n = 96, N_t = N_r = 2)$. We observe that cross-correlation values are at least $\approx$ 12 dB lower than the energy of each codeword $(n/(2N_E))$. This confirms that learned codewords belonging to different encoders are *nearly-orthogonal* to each other. Similarly, Fig. 4(b) shows the distribution of the positive entries (in dB) of c_{intra} (12) for the same codebooks plotted in Fig. 4(a). Note that the largest positive entry of c_{intra} is ≈ 2.5 dB lower than the energy of a codeword implying that the minimum L2-distance among the codewords from the same encoder is not insignificant.

B. Codebook Properties After a MIMO Channnel

The MIMO-NOS codebook $\{\boldsymbol{\mathcal{C}}_j\}$ exhibits the near-orthogonal property and reasonable minimum distances before MIMO transmission. This observation aligns with the results in [1], which only considers single antenna transmission cases.

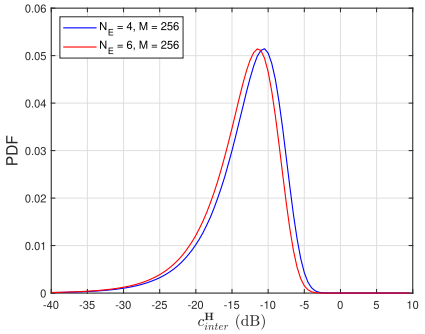


(a) *Inter*-correlation distribution before MIMO channels

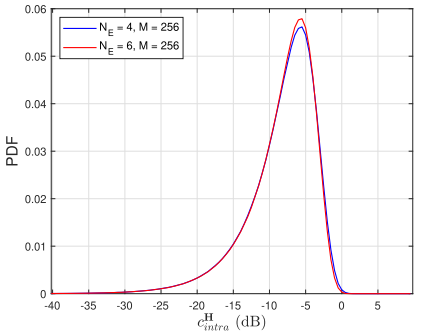


(b) *Intra*-correlation distribution before MIMO channels

Fig. 4. The distribution of the absolute value of entries (positive entries) in c_{inter} / c_{intra} for the two codebooks learned with ($N_E = 4, M = 256$, $n = 64, N_t = N_r = 2$) and ($N_E = 6, M = 256, n = 96, N_t = N_r = 2$).



(a) *Inter*-correlation distribution after MIMO channels



(b) *Intra*-correlation distribution after MIMO channels

Fig. 5. The distribution of the absolute values of $c^{\boldsymbol{H}}_{inter}$ entries and positive entries of $c^{\boldsymbol{H}}_{intra}$ using random channel realizations $\boldsymbol{H}$ in a 4×4 MIMO system.

In this section, we further inspect codebook properties after the MIMO channel.

For a random MIMO channel realization, $\boldsymbol{H}$, whose entries are independent zero-mean complex Gaussian with unit variance, the post-channel codebook is updated from (10) to

$$\boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}[m] = vec(\boldsymbol{H}\, f_{STC}(\boldsymbol{\mathcal{C}}_j[m])) \tag{13}$$

where $\boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}} \in \mathbb{C}^{N_r M_c \times M}$. Following the same principle in (11), the *inter*-correlation tensor $c^{\boldsymbol{H}}_{inter} \in \mathbb{R}^{N_E \times (N_E-1) \times M \times M}$ is obtained by:

$$c^{\boldsymbol{H}}_{inter}[i,j,k,l] = \frac{|\Re(\boldsymbol{\mathcal{C}}^{\dagger}_{\boldsymbol{i},\boldsymbol{H}}[k]\boldsymbol{\mathcal{C}}_{\boldsymbol{j},\boldsymbol{H}}[l])|}{N_r n/(2N_E)}$$
$$i,j \in [1,N_E]; i \neq j; k,l \in [1,M], \tag{14}$$

where the denominator $N_r n/(2N_E)$ is obtained by the expectation of $||\boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}[m]||_2^2$ over random realizations of $\boldsymbol{H}$:

$$\begin{aligned}\mathbb{E}(||\boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}[m]||_2^2) &= \boldsymbol{\mathcal{C}}_j^{\dagger}[m]\mathbb{E}((\boldsymbol{I}_{M_c} \otimes \boldsymbol{H})^{\dagger}(\boldsymbol{I}_{M_c} \otimes \boldsymbol{H}))\boldsymbol{\mathcal{C}}_j[m] \\ &= N_r n/(2N_E). \end{aligned} \tag{15}$$

In (15), $\boldsymbol{I}_{M_c}$ is a $M_c \times M_c$ identity matrix, $\otimes$ is the Kronecker product and $\mathbb{E}(\boldsymbol{H}^{\dagger}\boldsymbol{H}) = N_r \boldsymbol{I}_{N_t}$ holds. Similarly the *intra*-correlation, $c^{\boldsymbol{H}}_{intra}$ after the MIMO channel is defined by:

$$c^{\boldsymbol{H}}_{intra}[i,k,l] = \frac{\Re(\boldsymbol{\mathcal{C}}^{\dagger}_{i,\boldsymbol{H}}[k]\boldsymbol{\mathcal{C}}_{i,\boldsymbol{H}}[l])}{N_r n/(2N_E)}$$
$$i \in [1,N_E]; k \neq l; k,l \in [1,M]. \tag{16}$$

To obtain empirical distributions, we randomly instantiate one thousand 4×4 MIMO channel matrices $\boldsymbol{H}$'s and evaluate both $c^{\boldsymbol{H}}_{inter}$ and $c^{\boldsymbol{H}}_{intra}$ realizations. The distribution of absolute values of $c^{\boldsymbol{H}}_{inter}$ entries and the positive entries of $c^{\boldsymbol{H}}_{intra}$ are plotted in Fig. 5(a) and (b) respectively. The codebooks used for this evaluation are the same ones used in Fig. 4. Fig. 5(a) shows that the correlation between codewords belonging to different encoders after a MIMO channel is not negligible, and thus they do not preserve the *near-orthogonal* property any more. The maximum cross-correlation $c^{\boldsymbol{H}}_{inter}$ is only 4 dB lower than the expected energy of each received codeword ($N_r n/(2N_E)$) which is significantly higher than the maximum of the pre-channel correlation c_{inter} shown in Fig. 4(a). Meanwhile, $c^{\boldsymbol{H}}_{intra}$ shown in Fig. 5(b) has the largest positive element comparable to the expected codeword energy, implying significant minimum distance reduction between codewords from the same encoder after the MIMO channel.

IV. K-Best Assisted Decoding

Significant post-channel interference and codeword distance reduction observed in the previous section motivate the need for an efficient algorithm to mitigate these issues to attain close-to-ML decoding performance. Since the ML solution is practically infeasible due to excessive complexity, we propose and investigate a new practical CRC-assisted looped K-best tree-search algorithm for the learned MIMO-NOS code. Later, we will show that the proposed algorithm

significantly outperforms the neural network-based decoder which was used to train the learned MIMO-NOS codebook.

A. K-Best MIMO-NOS Decoding

The encoder and decoder neural network pair introduced in Section II is trained to minimize the number of bit errors per vector/codeword. However, typical mMTC applications do not tolerate any bit errors in a short packet, hence the primary objective of our scheme is to minimize the PER. For that, we include CRC bits in the information message to enhance the reliability of short packets in the low SNR regime. In our scenario, each transmitted block $\boldsymbol{S} \in \mathbb{C}^{N_t \times M_c}$ corresponds to a packet (which is obtained by space-time reshaping of a codeword, $\boldsymbol{S} = f_{STC}(\boldsymbol{s})$).

Consider the joint probability $P(\boldsymbol{x}_1^{m_1}, \cdots, \boldsymbol{x}_{N_E}^{m_{N_E}} | \boldsymbol{Y}, \boldsymbol{H})$ where $m_j \in [1, M]$. We desire to find the top-K (K-best) combinations that maximize the joint probability over all possible combinations of one-hot vectors $\{\boldsymbol{x}_1^{m_1}, \cdots, \boldsymbol{x}_{N_E}^{m_{N_E}}\}$. Note that in [1], we have solved the top-K searching problem of the learned NOS code in the single antenna AWGN channel. However, the assumption that the codewords are near-orthogonal *after* the channel is no longer valid for the MIMO transmission as shown in Fig. 5. Thus the joint probability does not factorize into products of marginal probabilities $\prod_{j=1}^{N_E} p(x_j^{m_j} | \boldsymbol{Y}, \boldsymbol{H})$ as in the single antenna AWGN channel case [1]. For the MIMO-NOS code, a procedure of finding the top-K combinations is proposed as follows.

The joint probability of a MIMO-NOS code follows the expression:

$$P(\boldsymbol{x}_1^{m_1}, \cdots, \boldsymbol{x}_{N_E}^{m_{N_E}} | \boldsymbol{Y}, \boldsymbol{H}) \propto \exp\{-\frac{1}{2N_t\sigma^2} ||\boldsymbol{y} - \sum_{j=1}^{N_E} \boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}[m_j]||_2^2\}, \quad (17)$$

where $\boldsymbol{y} \in \mathbb{C}^{N_r M_c \times 1}$ is the vectorized version of the received matrix $\boldsymbol{Y}$ and $\boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}$ is the codebook corresponding to $\boldsymbol{H}$ defined in (13). The problem of finding K candidates maximizing the joint probability is equivalent to finding m_j's that minimize the L2-distance $||\boldsymbol{y} - \sum_{j=1}^{N_E} \boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}[m_j]||_2^2$. It is practically infeasible for large N_E and M to identify the exact K-best candidates. To adopt the principle of K-best tree searching and pruning algorithms designed for near-ML MIMO detection [27], [28], we decompose the L2 term in (17), i.e., $||\boldsymbol{y} - \sum_{j=1}^{N_E} \boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}[m_j]||_2^2$ into four terms:

$$2\Re(\boldsymbol{\mathcal{C}}_{N_E,\boldsymbol{H}}^{\dagger}[m_{N_E}] \sum_{j=1}^{N_E-1} \boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}[m_j] - \boldsymbol{\mathcal{C}}_{N_E,\boldsymbol{H}}^{\dagger}[m_{N_E}]\boldsymbol{y}) + ||\boldsymbol{y} - \sum_{j=1}^{N_E-1} \boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}[m_j]||_2^2 + ||\boldsymbol{\mathcal{C}}_{N_E,\boldsymbol{H}}[m_{N_E}]||_2^2, \quad (18)$$

where the first term is the same as the LHS except the summation is from 1 to $N_E - 1$. To allow recursive metric evaluation, define the *score metric* $s^{(l)} = ||\boldsymbol{y} - \sum_{i=1}^{l} \boldsymbol{\mathcal{C}}_{i,\boldsymbol{H}}[m_i]||_2^2$, which can be expressed as:

$$s^{(l)} = s^{(l-1)} + ||\boldsymbol{\mathcal{C}}_{l,\boldsymbol{H}}[m_l]||_2^2 + 2\Re(\boldsymbol{\mathcal{C}}_{l,\boldsymbol{H}}^{\dagger}[m_l]\boldsymbol{u}^{(l-1)} - \boldsymbol{\mathcal{C}}_{l,\boldsymbol{H}}^{\dagger}[m_l]\boldsymbol{y}), \quad (19)$$

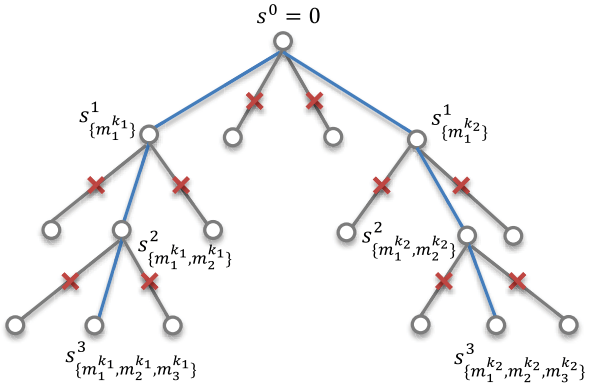


Fig. 6. The proposed K-best algorithm. The two blue branches indicate $K = 2$ survived paths in the tree.

where $\boldsymbol{u}^{(l-1)} = \sum_{i=1}^{l-1} \boldsymbol{\mathcal{C}}_{i,\boldsymbol{H}}[m_i]$ is the *cumulative vector*. Our objective is to find K-best candidates with the top-K smallest *score metric* $s^{(l)}$ for each l-th layer and prune all the other candidates using a tree structure shown in Fig. 6. We start from the root of the tree and initialize the score $s^{(0)} = 0$. For the k-th ($k \in [1, K]$) survived node in the $(l-1)$-th layer with accumulated indices $(m_1^k, \cdots, m_{l-1}^k)$, the metrics of all its children nodes with index $m_l \in [1, M]$ are calculated based on (19), satisfying

$$s^{(l)}_{m_1^k, \cdots, m_{l-1}^k, m_l} = s^{(l-1)}_{m_1^k, \cdots, m_{l-1}^k} + ||\boldsymbol{\mathcal{C}}_{l,\boldsymbol{H}}[m_l]||_2^2 + 2\Re(\boldsymbol{\mathcal{C}}_{l,\boldsymbol{H}}^{\dagger}[m_l]\boldsymbol{u}_k^{(l-1)} - \boldsymbol{\mathcal{C}}_{l,\boldsymbol{H}}^{\dagger}[m_l]\boldsymbol{y}), \quad (20)$$

where $\boldsymbol{u}_k^{(l-1)} = \sum_{i=1}^{l-1} \boldsymbol{\mathcal{C}}_{i,\boldsymbol{H}}[m_i^k]$. In this way, KM metrics are obtained and we only preserve the top-K candidates to serve as the survived parent nodes for the next layer whereas all the other candidates are pruned from the tree. We define this selection process as *SelectNodes*. By repeatedly extending and pruning the K-best tree, K survived paths are obtained at the last layer. The accumulated indices from the layer 1 to N_E of the k-th survived path are denoted as $(m_1^k, \cdots, m_{N_E}^k)$. By converting each m_j^k to a bit sequence $\boldsymbol{b}_j^k$ and concatenating them together, we obtain a bit sequence $\boldsymbol{b}^k$ for the subsequent CRC validation.

A well-known weakness of the K-best decoding algorithm is the error propagation. Any error made in previous layers can mislead the decisions in the following layers. To mitigate this issue, we follow the principle in [10] to first decode the vectors from $\{\boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}\}$ that are more 'reliable' based on the score metric calculated during the tree search by changing the decoding order of remaining layers in the tree. We denote this process as *ChooseLayer*. Two different sorting approaches are proposed in [10], namely, *per-layer* sorting and *per-branch* sorting. For *per-layer* sorting, we calculate the score metric $s^{(l)}$ assuming each of the remaining $(N_E - l + 1)$ layers as a possible l-th layer following (19) and using $\boldsymbol{u}^{(l-1)}$ of the up-to-now best candidate (with the smallest $s^{(l-1)}$). Then a layer with the minimum score metric is selected as the l-th layer to be processed next for all the K survivors. *Per-branch*

sorting also calculates the score metric $s^{(l)}$ of candidates for all remaining layers to determine the order. However, the layer evaluation is specific for each of the K survivors that has a unique cumulative vector $\boldsymbol{u}_k^{(l-1)}$. As a result, different survivors at each tree level may have distinct decoding orders. Since *per-branch* sorting determines a specific decoding order for each survivor, it has higher complexity, but it attains superior performance as each survivor can exploit a unique and better ordering for itself in general.

B. CRC-Assisted Looped K-Best Decoding

While *per-layer* and *per-branch* sorting approaches improve the error rate performance, any errors made in previous layers still cannot be corrected in the subsequent layers in the K-best decoding algorithm. To address that issue, we propose a *looped* K-best decoding algorithm that can correct errors in previously visited layers of the tree to further improve the PER performance.

The proposed looped K-best decoding algorithm performs N_{iter} additional layers of K-best decoding to revisit layers that were previously processed. After finishing regular K-best decoding for the final N_E-th layer, K survivors are obtained with corresponding accumulated indices $(m^k_{\pi_1}, \cdots, m^k_{\pi_{N_E}})$, the score metrics $s^{(N_E)}_{m^k_{\pi_1},\cdots,m^k_{\pi_{N_E}}}$, and the decoding order $(\pi_1, \ldots, \pi_{N_E})$.[3] To proceed to the next additional iteration of K-best decoding, it first updates K score metrics for these survivors by subtracting the terms that correspond to $\boldsymbol{\mathcal{C}}_{\pi_1,\boldsymbol{H}}[m^k_{\pi_1}]$ to obtain:

$$\begin{aligned}\tilde{t}^{(N_E+1)}_{m^k_{\pi_2},\cdots,m^k_{\pi_{N_E}}} &= s^{(N_E)}_{m^k_{\pi_1},\cdots,m^k_{\pi_{N_E}}} - ||\boldsymbol{\mathcal{C}}_{\pi_1,\boldsymbol{H}}[m^k_{\pi_1}]||_2^2 \\ &\quad - 2\Re(\boldsymbol{\mathcal{C}}^{\dagger}_{\pi_1,\boldsymbol{H}}[m^k_{\pi_1}] \sum_{j=2}^{N_E} \boldsymbol{\mathcal{C}}_{\pi_j,\boldsymbol{H}}[m^k_{\pi_j}] - \boldsymbol{\mathcal{C}}^{\dagger}_{\pi_1,\boldsymbol{H}}[m^k_{\pi_1}]\boldsymbol{y}),\end{aligned} \tag{21}$$

where $\tilde{t}$ denotes the updated metric. Then it repeats the standard process of the (revisited) first layer in the K-best decoding algorithm using the survived nodes as the parents by calculating the new score metrics of their children nodes with the index $m_{\pi_1} \in [1, M]$:

$$\begin{aligned}s^{(N_E+1)}_{m^k_{\pi_2},\cdots,m^k_{\pi_{N_E}},m_{\pi_1}} &= \tilde{t}^{(N_E+1)}_{m^k_{\pi_2},\cdots,m^k_{\pi_{N_E}}} + ||\boldsymbol{\mathcal{C}}_{\pi_1,\boldsymbol{H}}[m_{\pi_1}]||_2^2 \\ &\quad + 2\Re(\boldsymbol{\mathcal{C}}^{\dagger}_{\pi_1,\boldsymbol{H}}[m_{\pi_1}] \sum_{j=2}^{N_E} \boldsymbol{\mathcal{C}}_{\pi_j,\boldsymbol{H}}[m^k_{\pi_j}] - \boldsymbol{\mathcal{C}}^{\dagger}_{\pi_1,\boldsymbol{H}}[m_{\pi_1}]\boldsymbol{y}).\end{aligned} \tag{22}$$

One important aspect in the proposed looped K-best is that, among the newly generated KM candidates from the revisited layer, it only selects K *distinct* candidates with the best score metrics obtained with the updated accumulated indices $(m^k_{\pi_2}, \cdots, m^k_{\pi_{N_E}}, m^k_{\pi_1})$ and new ordering $(\pi_2, \cdots, \pi_{N_E}, \pi_1)$. These indices are reordered to $(m^k_1, \cdots, m^k_{N_E})$ which will be further converted into bit sequences. This process repeats for the next revisited layer until N_{iter} additional layers of K-best tree decoding are processed. Fig. 7 depicts the decoding process of the looped K-best decoding algorithm using an example with $(N_E = 3, M = 3, K = 2)$ and $N_{iter} = 2$.

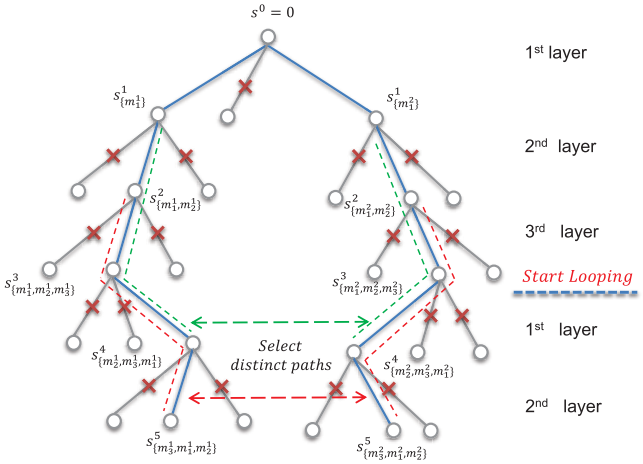


Fig. 7. The proposed looped K-best algorithm with parameter $(N_E = 3, M = 3, K = 2)$ with two additional iterations. The decoding order (π_1, π_2, π_3) is assumed to be $(1, 2, 3)$. The two blue branches indicate two survived paths in this example.

An interesting property of the proposed looped K-best decoding algorithm is that the score metrics of the K survivors are non-increasing with respect to N_{iter}. It is expected as we revisit the first element $m^k_{\pi_1}$ for the k-th survived path, it is always possible to choose the original element $m^k_{\pi_1}$ selected in the previous round, maintaining the same score metric. However, in many cases, the algorithm can find new paths with smaller score metrics to improve the performance.

We emphasize that the looped K-best needs a new constraint (which is unnecessary in the original K-best algorithm) to select *distinct* paths from KM candidates that have unique metrics (22) without duplication. We term this process as *SelectDistinctNodes* to distinguish it from the procedure *SelectNodes* previously defined. In the original K-best decoding without a loop, the K survivors from the first layer are always different although they might share the same path for the remaining $(N_E - 1)$ layers. One possible example is $(m^{(1)}_{\pi_1}, m_{\pi_2}, \cdots, m_{\pi_{N_E}})$ and $(m^{(2)}_{\pi_1}, m_{\pi_2}, \cdots, m_{\pi_{N_E}})$ as the final $K = 2$ candidates. In this case, when the first branch is revisited during the looped K-best decoding, it is likely that these two survived paths select the same m_{π_1} making the two paths identical and reducing the effective K from 2 to 1. To avoid such conditions, the proposed algorithm is constrained to only maintain *distinct* survivor paths by eliminating duplicated paths with the same score metric. For that, we first sort the KM score metrics in an increasing order and then eliminate duplicated metrics in the list before we select the final K best unique survivor metrics.

Once the algorithm finishes processing N_{iter} additional layers, K survived paths (after ordering them back to the original transmit order) are converted to K bit-sequences $\boldsymbol{b}^k$. Finally,

[3]Although we assume *per-layer* sorting for simplicity, it is straightforward to extend it to *per-branch* sorting.

Algorithm 1 CRC-Aided Looped K-Best Decoding Algorithm With *Per-Layer* Sorting

Input : $K, N_{iter}, \boldsymbol{y}, \{\boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}\}$
Output: decodedBits, errFlag

for $k = 1$ **to** K **to**
  $\boldsymbol{u}(k) \leftarrow 0$ (zero accumulative vector)
  $s(k) \leftarrow 0$ (zero score metric)
  $\mathsf{idx}(k) \leftarrow [\,]$ (empty candidate index)
$\mathcal{L} \leftarrow [\,]$ (empty decoded layer index)
for $j = 1$ **to** N_E **to**
  $l_j \leftarrow$ `ChooseLayer` $(\overline{\mathcal{L}})$
  $\mathcal{L} \leftarrow [\mathcal{L}, l_j]$
  for $k = 1$ **to** K **to**
    $\mathbf{s}^{tmp}(k) \leftarrow s(k) - 2\Re(\boldsymbol{y}^{\dagger}\boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}} - \boldsymbol{u}^{\dagger}(k)\boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}}) + diag(\boldsymbol{\mathcal{C}}^{\dagger}_{l_j,\boldsymbol{H}}\boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}})$
  $[\boldsymbol{s}, \mathsf{idx}_{new}, \mathsf{anc}] \leftarrow$ `SelectNodes` $(\boldsymbol{s}^{tmp}, K)$
  for $k = 1$ **to** K **to**
    $\boldsymbol{u}(k) \leftarrow \boldsymbol{u}(\mathsf{anc}(k)) + \boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}}[\mathsf{idx}_{new}(k)]$
    $\mathsf{idx}(k) \leftarrow [\mathsf{idx}(\mathsf{anc}(k)), \mathsf{idx}_{new}(k)]$
for $j = 1$ **to** N_{iter} **to**
  $l_j, \mathsf{idx}_j \leftarrow \mathcal{L}(j), \mathsf{idx}(:, j)$
  $\mathcal{L} \leftarrow [\mathcal{L}, l_j]$
  for $k = 1$ **to** K **to**
    $\boldsymbol{u}(k), I_{jk} \leftarrow \boldsymbol{u}(k) - \boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}}[\mathsf{idx}_j(k)], \mathsf{idx}_j(k)$
    $t^{tmp}(k) \leftarrow s(k) + 2\Re(\boldsymbol{y}^{\dagger}\boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}}[I_{jk}] - \boldsymbol{u}^{\dagger}(k)\boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}}[I_{jk}]) - ||\boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}}[I_{jk}]||_2^2$
    $\boldsymbol{s}^{tmp}(k) \leftarrow t^{tmp}(k) - 2\Re(\boldsymbol{y}^{\dagger}\boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}} - \boldsymbol{u}^{\dagger}(k)\boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}}) + diag(\boldsymbol{\mathcal{C}}^{\dagger}_{l_j,\boldsymbol{H}}\boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}})$
  $[\boldsymbol{s}, \mathsf{idx}_{new}, \mathsf{anc}] \leftarrow$ `SelectDistinctNodes` $(\boldsymbol{s}^{tmp}, K)$
  for $k = 1$ **to** K **to**
    $\boldsymbol{u}(k) \leftarrow \boldsymbol{u}(\mathsf{anc}(k)) + \boldsymbol{\mathcal{C}}_{l_j,\boldsymbol{H}}[\mathsf{idx}_{new}(k)]$
    $\mathsf{idx}(k) \leftarrow [\mathsf{idx}(\mathsf{anc}(k)), \mathsf{idx}_{new}(k)]$
$\mathsf{idx}, \mathcal{L} \leftarrow \mathsf{idx}(:, N_{iter} : end), \mathcal{L}(N_{iter} : end)$
outputList $\leftarrow$ `Reorder` (idx, $\mathcal{L}$)
while errFlag $\neq 0$ *and* $k \leq K$ **do**
  decodedBits $\leftarrow$ `IdxToBits` (outputList (k))
  errFlag $\leftarrow$ `CRCDecode` (decodedBits)

we pass them to check the CRC bits for error detection. A candidate $\boldsymbol{b}^k$ with a smaller metric is checked first until one that passes the CRC bits is identified as the final decoding output. The entire CRC-assisted looped K-best decoding algorithm for the learned MIMO-NOS code is summarized in Algorithm 1. Note $diag(\boldsymbol{A})$ in Algorithm 1 is a function that returns the diagonal elements $(a_{11}, a_{22}, \cdots, a_{NN})$ of an $N \times N$ square matrix $\boldsymbol{A}$.

V. Evaluation

The PER performance of the proposed scheme is evaluated via Monte-Carlo simulations. For short MIMO message transmission, we compare the performance of the learned MIMO-NOS coding using the CRC-assisted looped K-best decoding algorithm with a polar-coded MIMO-QPSK (quadrature phase shift keying) scheme demodulated/decoded by ML MIMO detection and CRC-assisted list polar decoding. We also compare the performance of the proposed looped K-best decoding with the neural network-based NOS decoder that is used to train/learn the NOS codebook. The decoding complexity and latency for the proposed looped K-best decoder are analyzed, and we also carry out experiments to quantify the gain from different components of the proposed MIMO-NOS scheme. To improve the readability of the simulation results, we summarize the key parameters defined in Sections II, III and IV in Table I.

TABLE I
List of Important Variables

Variable	Description
N_E	Number of encoder-decoder pairs used in the MIMO-NOS scheme.
M	Dimension of each one-hot input vector, $\boldsymbol{x}_j$, to an encoder.
n	Length of the real-valued codeword $\boldsymbol{s}$ in (4).
R	The code rate of MIMO-NOS encoding, $R = \frac{N_E \log_2(M)}{n}$.
N_t	Number of transmit antennas.
N_r	Number of receive antennas.
N_t^l	The number of transmit antennas during training.
N_r^l	The number of receive antennas during training.
K	The number of survivors kept in each layer of the K-best tree search.
N_{iter}	The number of loops in the looped K-best decoding algorithm.
N_{CRC}	The number of CRC bits appended to the information bits.
L	List size for the list decoding polar.

A. Deep Learning Model Training

The neural network structure shown in Fig. 2 is defined by the parameter set $(N_E, M, n, N_t, N_r, T_1, T_2)$ where T_1 denotes the number of hidden neurons in the encoder F^j_{enc} and decoder F^j_{dec}, $j \in [1, N_E]$, and T_2 is the number of hidden neurons in the residual connection module Res. We set $T_1 = 4n, T_2 = 128$ for all experiments. All DNN models are trained for 5×10^3 epochs with 5×10^5 training samples (packets or codewords) for each epoch. During training, each training sample/packet observes an independent realization of the random MIMO channel matrix $\boldsymbol{H} \in \mathbb{C}^{N_r \times N_t}$ as described in Section II. The batch size is set to 1024 and the dynamic learning rate changes linearly from the initial value of 2×10^{-4} to the final 2×10^{-6}. All models are trained under a fixed SNR of 10 dB although they are evaluated under different mismatched SNRs. Once the deep learning model training is complete, we construct a lookup table (LUT) of the learned codebook, $\{\boldsymbol{\mathcal{C}}_j\}$ as defined in (10).

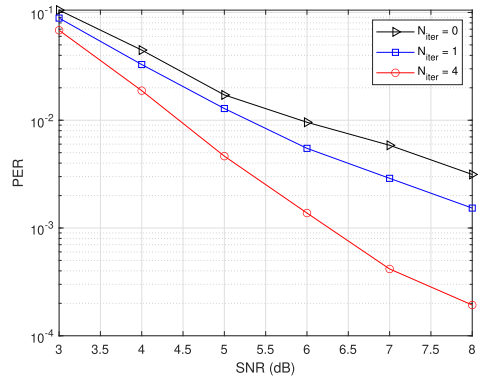


Fig. 8. PER performance of the CRC-assisted looped K-best decoder for the learned MIMO-NOS codebook trained with the system parameter set $(N_E = 4, M = 256, n = 64, N_t = N_r = 4)$. We set $K = 16$ and $N_{CRC} = 11$. The number of additionally processed layers for the K-best decoder is set to N_{iter}.

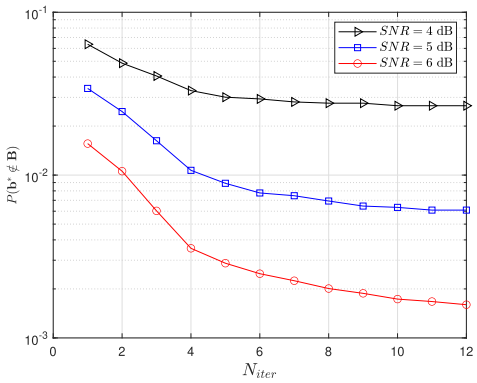


Fig. 9. Error rate performance of the looped K-best algorithm with various N_{iter} settings given the parameter set $(N_E = 6, M = 256, n = 96, N_t = N_r = 4)$ under different SNRs.

B. Performance of the Looped K-Best Decoder

For PER evaluation, each packet goes through an independent MIMO channel $\boldsymbol{H}$ while the channel stays the same for a single packet. Fig. 8 shows the performance of the CRC-assisted looped K-best decoder given the system parameter set of $(N_E = 4, M = 256, n = 64, N_t = N_r = 4)$. This corresponds to transmitting 32 $(= N_E \cdot \log_2(M))$ information bits (including CRC bits) with 4 transmit (N_t) and receive (N_r) antennas with 8 $(M_c = n/(2N_t))$ MIMO channel uses. In Fig. 8, K is 16, the number of CRC bits N_{CRC} is 11, and N_{iter} denotes the number of additional layer decoding iterations. $N_{iter} = 0$ corresponds to the original K-best decoding without any loop. Relatively worse performance of $N_{iter} = 0$ is expected since the errors made in earlier layers can not be corrected without additional loops. The looped K-best algorithm with a higher N_{iter}, on the other hand, can correct some previous errors and it attains a 1.5 dB gain with $N_{iter} = 4$ for PER $\approx 10^{-2}$.

We then evaluate the error rate performance of the looped K-best algorithm with respect to a wide range of N_{iter} in Fig. 9 for MIMO-NOS code with the parameter set $(N_E = 6, M = 256, n = 96, N_t = N_r = 4)$ and evaluated at different SNRs. We set $K = 16$ and $N_{CRC} = 11$. The error rate performance is quantified using the probability $P(\boldsymbol{b}^* \notin \boldsymbol{B})$ where $\boldsymbol{b}^*$ is the correct bit sequence and $\boldsymbol{B}$ is the set of K-best candidates $\boldsymbol{b}^k, k \in [1, K]$, obtained by the algorithm. As N_{iter} increases, $P(\boldsymbol{b}^* \notin \boldsymbol{B})$ monotonically decreases resulting in the improved PER. Fig. 9 further shows that the error rate performance improvement from the increased number of iterations is more substantial when the SNR is higher. To strike a good balance between the PER performance and the decoding complexity, we set $N_{iter} = N_E$ for the remaining evaluations unless noted otherwise.

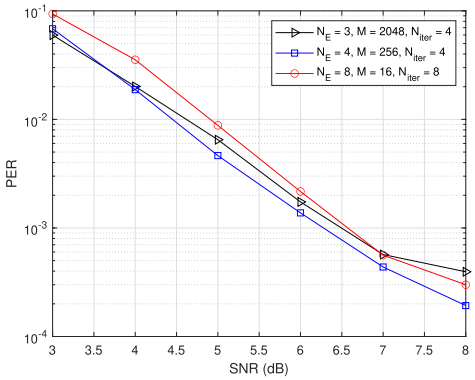


Fig. 10. PER performance of different (N_E, M) combinations with ≈ 32 information bits for a 4×4 MIMO system. There is an optimal N_E for the target rate as the performance is not a monotonic function of N_E given the target rate.

C. Performance With Different System Parameters

As shown in Table I, the rate R of the proposed MIMO-NOS scheme is determined by the number of information bits $(N_E \cdot \log_2(M))$ and the length of the complex-valued codeword $(n/2)$, satisfying $R = \frac{N_E \log_2(M)}{n/2}$. With a fixed n, there are different (N_E, M) combinations to obtain the same target rate R whereas one configuration outperforms the other. Our prior work [1] for single antenna AWGN channel argues that the number of superimposed vectors N_E should be minimized (with a larger M) as long as the complexity (i.e., model size) of the neural network to learn a NOS codebook is manageable. However, we find that for the proposed MIMO-NOS coding, using a smaller N_E (and larger M) does not necessarily improve the PER performance while it definitely increases the complexity of the network model. The analysis is involved but numerical evaluation of the score metric in (20) under the MIMO channel shows that there is an optimal N_E (and corresponding M) that balances the inter- and intra-codeword correlation tradeoff. Fig. 10 shows the PER performance of three different (N_E, M) combinations that are $(N_E = 3, M = 2048)$, $(N_E = 4, M = 256)$ and $(N_E = 8, M = 16)$ evaluated under 4×4 MIMO transmission with $n = 64$, $K = 16$, and $N_{CRC} = 11$. Note that all these settings have (almost) the same rate. The setting of $(N_E = 4, M = 256)$ outperforms the other with smaller or larger N_E's. Note that N_{iter} is set to 4 for $(N_E = 3, M = 2048)$ setting which has a larger FLOP count and decoding latency compared with the $N_E = 4$ setting with the same N_{iter}. We set $N_{iter} = 8$ for $(N_E = 8, M = 16)$ setting which has a similar FLOP count but larger decoding

latency compared to $N_E = 4$ setting with $N_{iter} = 4$. As can be shown in Fig. 10, the $N_E = 4$ setting outperforms the others with a less/similar FLOP count and shorter decoding latency. We also observed that $(N_E = 4, M = 256)$ outperforms the other settings when all use unlimited N_{iter}. It is worth noting that the $(N_E = 8, M = 16)$ setting is inferior to $(N_E = 3, M = 2048)$ at low SNRs while the opposite is observed at high (> 7 dB) SNRs. It is because of the tradeoff between inter- and intra-codeword distances that the proposed K-best algorithm experiences during the decoding process. A larger N_E (smaller M) creates more severe inter-codeword interference with a deeper tree structure that makes the algorithm suffer from early decoding errors in the tree at low SNRs. When the SNR is relatively high with lower chance of early stage errors in the K-best decoding, the performance is limited by the intra-codeword distance as more candidates M are evaluated for each layer. Although it is difficult to accurately analyze this tradeoff, Fig. 10 shows that there is an optimal parameter set and the PER performance is not necessarily a monotonic function of N_E or M. Empirically, we observed that a setting with $M = 256$ usually outperforms others (as observed in Fig. 10). Hence, we use $M = 256$ (with a corresponding N_E to attain the target rate) for the rest of the paper to evaluate the performance of the proposed MIMO-NOS scheme.

In the proposed scheme, the dimension of the codebook $\{\boldsymbol{\mathcal{C}}_j\}$ (10) is determined by the parameter set (M, n) and it does not depend on the MIMO configuration, (N_t, N_r). For a given codebook $\{\boldsymbol{\mathcal{C}}_j\}$, the MIMO configuration (N_t, N_r) defines the space time coding scheme by reshaping the samples of a transmitted codeword with proper space and time indices as discussed in Section II-B. This implies that it is possible to use a codebook for different MIMO settings by simple reshaping even though they are not necessarily identical to that used during the codebook training. In other words, one can apply reshaping based on the desired (N_t, N_r) to an existing learned codebook trained with different (N_t, N_r) as long as (N_E, M, n) is unchanged. To facilitate the discussion to follow, we distinguish the number of transmit and receive antennas used during the training by N_t^l and N_r^l, respectively. Consequently, N_t and N_r denotes the number of antennas for evaluation of a learned MIMO-NOS codebook. We observed that N_r^l makes little impact to the PER performance of the codebook for a given evaluation setup N_t or N_r as long as $N_r^l \geq N_t^l$ holds. Thus we only show the impact of $N_t^l (= N_r^l)$ in the following discussion.

Fig. 11 shows the PER performance of the codebooks for the setting $(N_E = 6, M = 256, n = 96)$ trained with $N_t^l = N_r^l = 2, 3, 4$ and evaluated for $N_t = N_r = 4$ MIMO transmission. We set $K = 16$, $N_{CRC} = 11$ and $N_{iter} = 4$. Intuitively, one would expect the best performance when $N_t^l = N_t$. However, the simulation shows that the codebooks trained with $N_t^l = 2$ or 3 outperform the one with $N_t^l = 4$ for the $N_t = 4$ evaluation, showing the 'mismatch' between N_t^l and N_t for the optimal performance.

To understand this mismatch, Fig. 12 analyzes inter-correlation $c_{inter}^{\boldsymbol{H}}$ and intra-correlation $c_{intra}^{\boldsymbol{H}}$ for different N_t^l's with random 4×4 MIMO channel realizations. Notice that

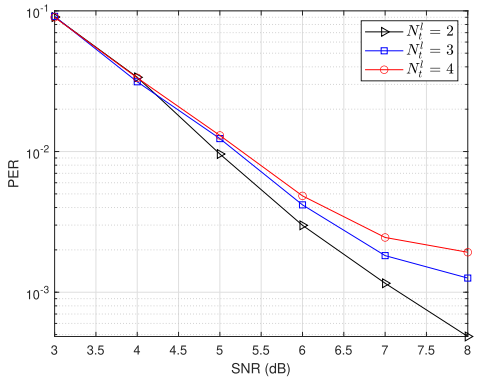
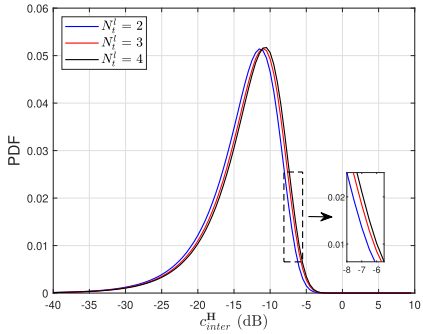
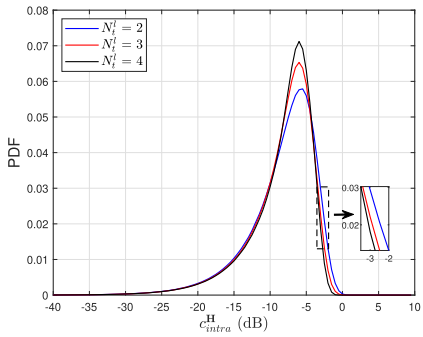


Fig. 11. The PER performance of the codebooks learned under $N_t^l = 2, 3, 4$ applied to the 4×4 MIMO transmissions with parameters $(N_E = 6, M = 256, n = 96, K = 16, N_{iter} = 6, N_{CRC} = 11)$.



(a) *Inter*-correlation distribution



(b) *Intra*-correlation distribution

Fig. 12. *Inter*- and *Intra*-correlation distributions for codebooks learned under different transmit antennas $N_t^l = 2, 3, 4$ evaluated for 4×4 MIMO transmission given the system parameter $(N_E = 6, M = 256, n = 96)$.

the codebooks trained with N_t^l =2 or 3 have better $c_{inter}^{\boldsymbol{H}}$ distribution compared with the $N_t^l = 4$ counterpart, while the $N_t^l = 4$ codebook has better $c_{intra}^{\boldsymbol{H}}$ distribution. From this experiment using the given parameter set, we observe that the PER of the proposed looped K-best decoding is dominated by the inter-codeword interference that propagates down to later tree levels during the looped K-best decoding. When inter-codeword interference is correctly cancelled out, decoding of each layer (whose performance is governed by intra-codeword correlation) using a reasonably high $K \gg 1$ with respect to M does not limit the PER performance at high SNRs. The codebook learned with $N_t^l = 2$ strikes

the balance between inter- and intra-codeword correlation for $N_t = 4$ evaluation.[4]

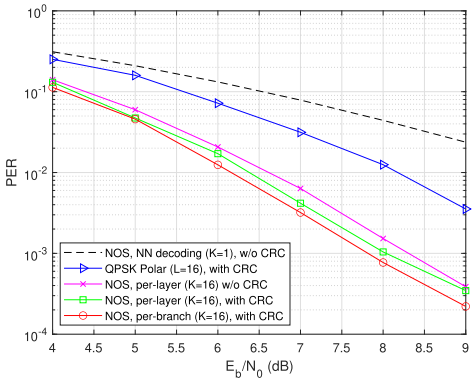
The above observation brings one question why the proposed MIMO-NOS framework learns a better codebook under a mismatched MIMO scenario $N_t^l \neq N_t$. It can be explained by the mismatch between the hand-crafted looped K-best decoder used for evaluation and the neural network-based one-shot decoder used for training as introduced in Section II. Although the looped K-best coding outperforms the neural network based decoder (as shown in the next subsection), it is not differentiable and thus cannot be directly used as a decoder for the end-to-end training to learn a codebook. Since the training is performed with a sub-optimal neural network-based decoder, the property of the learned codebook is not necessarily optimal for the proposed looped K-best coding algorithm.

An intuition is that it might be possible to learn a better MIMO-NOS codebook if the encoder was optimized with a neural network decoder that has a built-in K-best decoding structure. This resembles a prior work [29] in machine language translation where a (K) beam-searching algorithm is used to improve the BLEU score of the translation model. In our case, it would require selecting K survivors from KM number of branches and back-propagating the gradient through a $max_k(\cdot)$ operation (which is not differentiable). Devising and successfully training such a new neural network structure with a built-in K-best selection mechanism is non-trivial, thus left as future work.

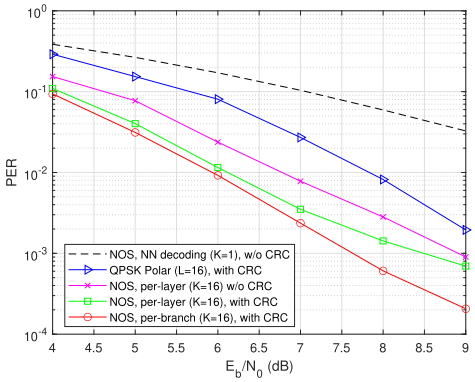
D. PER Performance Comparison With a Conventional Scheme

Finally, we compare the performance of our MIMO-NOS scheme with the conventional polar-coded MIMO system. As discussed in [5], the CRC-assisted polar code is proven to be robust for short packet transmission, thus we selected it as the baseline. Although there exist multiple computationally-efficient MIMO detection algorithms such as sphere decoding and K-best decoding [28], [30] that provide soft decisions, we choose the ML MIMO detection for the baseline to avoid degrading the polar code performance. We apply a successive cancellation list decoding algorithm (SCL) [5] with list size L to polar decoding.

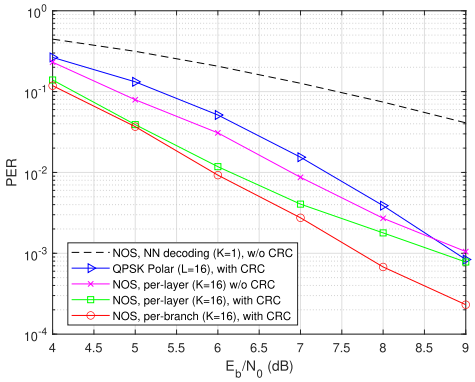
For comparisons with the CRC-assisted list decoding polar, we train the MIMO-NOS codebook with $N_t^l = N_r^l = 2$, and evaluate both in the 4×4 MIMO configuration. In the first case, we evaluate transmission of 32 message bits (Fig. 13(a)), and in the second and third case we increase the message length to 48 bits (Fig. 13(b)) and 64 bits (Fig. 13(c)). The MIMO-NOS scheme uses the parameter set $(N_E = 4, M = 256, n = 64)$ for the first case (32 bits), $(N_E = 6, M = 256, n = 96)$ for the second (48-bit), and $(N_E = 8, M = 256, n = 128)$ for the last case (64-bit) while $K = 16$ for all these cases. The baseline uses 3GPP polar code [31] with QPSK modulation and 0.5 coding rate for all these cases. Its list decoding size L is set to be $L = K = 16$ for fair comparison. The 11-bit CRC with a generator polynomial $x^{11}+x^{10}+x^9+x^5+1$ is adopted to both MIMO-NOS and the Polar code baseline. Note that for the proposed MIMO-NOS scheme, we also provide results using neural network decoder (introduced in Section II) with $K = 1$ without CRC and the looped K-best decoder with $K = 16$ without CRC to quantify the gain from different decoding algorithms and the CRC. Note that different schemes have different spectral efficiency depending on appending the CRC bits or not. Thus, for a fair comparison, we use E_b/N_0 as x-axis. The E_b/N_0 in our setup is defined as: $E_b/N_0 = SNR + 10\log_{10}(\frac{n/2}{N_E\log_2(M)-N_{CRC}})$, where SNR is shown in (6).



(a) 32 info bits, $(N_E = 4, M = 256, n = 64)$



(b) 48 info bits, $(N_E = 6, M = 256, n = 96)$



(c) 64 info bits, $(N_E = 8, M = 256, n = 128)$

Fig. 13. The proposed MIMO-NOS scheme outperforms the polar with ML MIMO detection under different number of information bits ranging from 32 to 64 bits in 4 × 4 MIMO channels.

[4]We note that the simulation results in Fig. 8 and Fig. 9 also adopt $N_t^l = N_r^l = 2$.

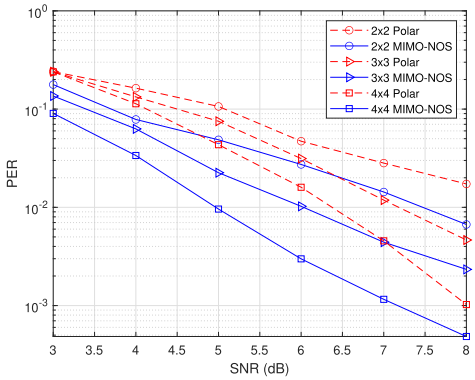


Fig. 14. The comparison of the MIMO-NOS and the baseline polar code applied to $2 \times 2, 3 \times 3$ and 4×4 MIMO transmissions given the parameters $(N_E = 6, M = 256, n = 96)$, $K = L = 16$, and $N_{CRC} = 11$. The information length is 48-bits.

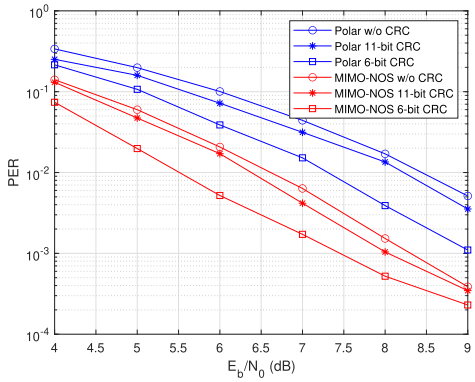


Fig. 15. The PER performance for the Polar baseline and proposed MIMO-NOS scheme with different numbers of CRC bits (11-bit, 6-bit, and 0-bit (no CRC)) over a 4×4 MIMO channel. The packet is 32 bits including information and CRC bits.

Fig. 13 presents the MIMO-NOS schemes with CRC (adopting either *per-branch* or *per-layer* sorting) outperform the polar baseline by $1-2$ dB in terms of E_b/N_0 for short messages in the range of 32 – 64 bits. We also observe the schemes adopting looped K-best decoder consistently outperform the scheme employing a neural network decoder labeled 'NOS, NN decoding ($K = 1$) w/o CRC' showing the effectiveness of the proposed decoding algorithm. It is due to the fact that, in the DNN-based decoding algorithm, each F_{dec}^{j} only estimates the posterior probability $p(x_j|\boldsymbol{y}, \boldsymbol{H})$ in a one-shot manner and searches one x_j which maximizes the posterior. Since each F_{dec}^{j} only selects one candidate, x_j, it cannot fully exploit the potential gain from the additional CRC bits (it quits when the only candidate does not pass the CRC check). Our CRC-aided looped K-best decoding algorithm, on the other hand, aims to find K combinations that maximize the joint posterior probability, $p(x_1, \cdots, x_{N_E}|\boldsymbol{y}, \boldsymbol{H})$. The selected K combinations are then fed to the CRC decoder to find the first one that passes CRC. Although adding CRC bits increases E_b, the improved PER performance offsets that overhead, providing gains in terms of E_b/N_0. We further note that the neural network for 'NOS, NN decoding ($K = 1$) w/o CRC' curve is trained and tested with the residual connection Res in parallel with the conventional MMSE detection. The importance of the Res module should be emphasized where a ≈ 2 dB gain is observed compared to a version without Res in the $N_t = N_r = 2$ setting. We can also observe the gain of the *per-branch* sorting over the *per-layer* sorting improves with the number of information bits from approximately 0.2 dB for 32 bits to 1 dB for 64 bits. Fig. 13(c) shows that the gain of the proposed scheme over the polar baseline reduces with a larger number of information bits, which is expected because polar coding is capacity-achieving when the codeword length is sufficiently long.

We now compare the performance of the MIMO-NOS scheme and the baseline under different MIMO settings. The parameters of the MIMO-NOS for this simulation are $(N_E = 6, M = 256, n = 96)$, 48 information bits, and $K = 16$ with *per-layer* sorting. The codebook is learned with $N_t^l = 2$. The SCL decoding based polar scheme has the same information bit length and coding rate of 0.5 with QPSK modulation and $L = 16$. Both schemes adopt 11-bit CRC and are tested with 2×2, 3×3, and 4×4 MIMO configurations. Fig. 14 shows the proposed MIMO-NOS scheme outperforms the polar baseline for all tested MIMO settings.

Finally, we study the impact of using different CRC lengths. Similar with the simulations in Fig. 13, our evaluation is based on E_b/N_0 which includes the energy overhead to transmit different number of CRC bits. In this experiment, the PER performance for both the Polar baseline and the proposed MIMO-NOS scheme using the aforementioned 11-bit CRC, a 6-bit CRC[5] and 0-bit CRC (without CRC) are simulated. We consider transmitting 32 bits (including CRC bits) over a 4×4 MIMO channel. The Polar baseline adopts QPSK modulation, ML MIMO detection, and list size $L = 16$ while we set $K = 16, N_{iter} = 4$ for the proposed looped K-best decoder. The comparison of these schemes is shown in Fig. 15. As can be seen from the figure, both the Polar baseline and the proposed MIMO-NOS scheme perform better using the 6-bit CRC compared with the 11-bit CRC specified in the 5G standard. Whereas it is observed that transmission without CRC bits has worse performance than using 6-bit or 11-bit CRC. For the same CRC setting, the proposed MIMO-NOS scheme outperforms the Polar baseline.

E. Complexity Analysis

We analyze the complexity of the *per-layer* sorting which can be estimated by summing the number of FLOPs (floating point operations) of the three parts: 1) constructing/updating the codebook for the observed MIMO channel realization, 2) initial K-best decoding, and 3) additional looped K-best decoding. To simplify the analysis, we assume $N_t = N_r = N$.

1) *Codebook Update*: We generate the post-channel codebook $\{\boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}\}$ from the original codebook $\{\boldsymbol{\mathcal{C}}_j\}$ by matrix multiplications (13) and calculate the norm of each codeword in $\{\boldsymbol{\mathcal{C}}_{j,\boldsymbol{H}}\}$. These steps are based on the QR decomposition of the channel CSI, $\boldsymbol{H}$, and they require $N_E M N(N+1)n/4$ and $N_E M n/2$ operations, respectively.

[5]The polynomial for the 6-bit CRC is $x^6 + x^5 + 1$ used in the 5G standard.

2) *Initial K-best Decoding*: This involves four sub-steps to choose the next layer (*ChooseLayer*), calculate metrics, select survivor nodes (*SelectNodes*) and update *cumulative vector*. At layer j, each sub-step requires $(N_E-j+1)Mn/2+(N_E-j+1)M, Kn/2+KMn/2+KM, 2KM+2(KM+1)H_{KM}-2(KM+3-K)H_{KM+1-K}-6K+6$, and $Kn/2$ operations, respectively where H_t is the t-th harmonic number. In the third sub-step, we use partial quick sort [32] to select K survivor nodes from KM candidates (denoted as $Choose(K, KM)$ for convenience).

3) *Looped K-best Decoding*: This step differs from the initial K-best decoding only in the first and third sub-step, which requires $3Kn/2+2K$ and $Choose(aK, KM)$ operations, respectively. The parameter $a = 1.5$ is chosen based on numerical evaluations to find a reasonable tradeoff between complexity and performance.

Summing these three steps, the total number of operations of the proposed algorithm is upper bounded by: $N_EMN(N+1)n/4+N_EMn/2+N_E(N_E+1)Mn/4+N_E(N_E+1)M/2+N_E(Kn+KM+KMn/2+Choose(K, KM))+N_{iter}(1.5Kn+2K+KMn/2+KM+Choose(1.5K, KM))-Kn$.

To evaluate the complexity of the baseline scheme, we use ML QPSK (Q = 4) MIMO detection where the total number of operations can be estimated as $NM_c\log_2 Q2^{N\log_2 Q}(0.5\,N^2+2.5\,N+1)$. For polar code list decoding, we use the result in [33] and estimate 10^3 FLOPs per bit when $L = 16$. Thus, for an example where $(N_E, M, n) = (4, 256, 64), K = 16, N_{iter} = 4$ and $N_t = N_r = N = 4$, the proposed MIMO-NOS scheme exhibits substantially higher complexity (1.569×10^6 FLOPs) compared to the QPSK Polar baseline (2.88×10^5 FLOPs).

However, we observed that the decoding speed of the proposed scheme can be faster than that of the baseline. This is because our K-best decoding algorithm is inherently parallel whereas the list decoding Polar uses successive cancellation which is sequential and unable to parallel hardware computation resources. The run time measured on Intel Xeon(R) Silver 4110 CPU using Python / Matlab implementations shows that the runtime of the proposed algorithm is approximately 2 times faster for the same aforementioned scenario.

F. Quantified Gain From Each Component

To quantify the gain from each component in the proposed framework, we show in Fig.16 the performance of our proposed scheme when each component was replaced by an alternative approach. In this experiment, the number of information bits (including CRC bits with $N_{CRC} = 11$) is set to be 32, MIMO configuration is set to 4×4, and we set $K = 16, N_{iter} = 4$ for the looped K-best decoder with *per-layer* sorting. The case 'Random codebook' is when the learned encoders are replaced with a randomly generated codebook $\boldsymbol{\mathcal{C}}_j[m]\sim\mathcal{N}(0, \boldsymbol{I}_{n/2}); j\in[1, N_E]; m\in[1, M]$ under the same power constraint, while the other components are kept the same to isolate the gain from the learned encoding/modulation. The gap from the proposed scheme ('Proposed $N_{iter} = 4$') can be explained by the lack of the near-orthogonal property discussed in Section III-A. A randomly generated codebook for a short block length does not satisfy a similar near-orthogonal property, thus its PER performance is limited by the interference between codewords.

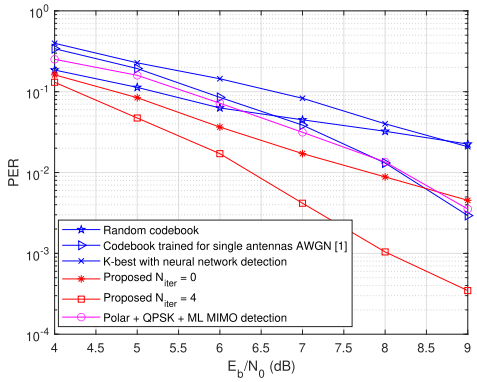


Fig. 16. The gain of the proposed scheme ('Proposed $N_{iter} = 4$') compared to the Polar baseline and alternative schemes replacing one component/parameter in the proposed scheme with another.

Fig. 16 shows the performance when the codebook is replaced with the one in [1] which is optimized for the single antenna AWGN channel (without the space-time mapping module). Although the codebook in [1] satisfies the near-orthogonal property at the transmitter, it does not consider the inter-codeword distance/orthogonality at the receiver after the MIMO channel. Therefore, it exhibits a significant performance gap compared with the proposed scheme where the codebook is trained with a space-time mapping module under the MIMO channel.

The codebook training of the proposed scheme involves a neural network decoder and residual-assisted MMSE MIMO detection as explained in Section II. We quantify the performance of the case when the explicit residual-assisted MMSE MIMO detection and neural network decoders are used in the K-best decoding algorithm introduced in [1]. Specifically, in the l-th layer, the K-best decoding is performed by selecting K candidates with the top-K values $\prod_{i=1}^{l}\boldsymbol{p}_i[m_i]$ where $\boldsymbol{p}_i$'s are obtained from the neural network decoders that take residual-assisted MMSE MIMO detection results as the input. For the $(l+1)$-th tree level, the K survivors in the l-th layer will be served as the parent nodes and we select K candidates with the top-K metrics among their children nodes while pruning the others. The K-best decoding process is repeated until it reaches the last layer. We set $K = 128$ for this neural network decoder-based scheme and compare it with the proposed looped K-best decoder with $K = 16$. As can be seen in Fig. 16, there is a significant gain from the proposed looped K-best decoder over the neural network decoder-based approach using the same codebook and space-time mapping. Fig. 16 also shows the gain of the proposed scheme when the number of iterations increases from 0 to 4.

G. Discussion

In this subsection, we will discuss the use-case for the proposed MIMO-NOS scheme and its limitations which invite

future studies. Our main use-case scenario is an uplink network from many devices (such as vehicles and factory robots) to a central powerful gateway (infrastructure) to exchange short control-type messages. These devices as well as the gateway may employ a modest number of antennas (2 – 4). Note that the overhead for MIMO CSI estimation at the receiver would be significant but using multiple antennas can still reduce the overall latency as the message payload length decreases by a factor of 2 – 4. The devices use a relatively simple MIMO-NOS encoder while the gateway uses the CRC-assisted looped K-best decoding algorithm on a parallel computing platform to reduce the latency. The superior PER performance as well as the lower latency property make the proposed MIMO-NOS scheme a promising scheme.

We then point out the limitations of the proposed scheme: As shown in Fig. 13, the performance gap between the learned MIMO-NOS and the polar baseline reduces as the message length increases. This can be explained using the inter-correlation of the codewords belonging to different codebooks after the MIMO channel, $c_{inter}^{\boldsymbol{H}}$ in (14). When the N_E grows larger for longer block lengths (with M fixed at 256), the error propagation problem of the proposed looped K-best decoding will become more severe unless $c_{inter}^{\boldsymbol{H}}$ is greatly reduced with the increased N_E. However, Fig. 5(a) indicates there is no significant improvement of the $c_{inter}^{\boldsymbol{H}}$ with a larger N_E, illustrating that the error propagation problem will definitely degrade the performance of the looped K-best decoder. With the analysis above, it is not practical to scale the proposed scheme to an arbitrarily long length although conventional superposition codes are known to be capacity achieving for long sequences [11]. Nevertheless, the proposed MIMO-NOS is a promising solution for reliable short message MIMO transmission in the low SNR regime with superior PER performance and an efficient decoding algorithm. Investigating new network structures and corresponding training schemes for learned superposition coding that scales better to longer information bit lengths is left as future work.

VI. Conclusion

This paper proposes a novel deep learning based MIMO-NOS coding scheme for reliable transmission of short messages in MIMO channels. The proposed end-to-end framework enables the encoder to successfully learn near-orthogonal superposition codewords with the aid of a neural network decoder. To improve the error rate performance, we propose and evaluate a CRC-assisted looped K-best decoder, which significantly outperforms the neural network decoder used during the training. We characterize the proposed MIMO-NOS coding and provide empirical evaluations with different MIMO settings and NOS encoding parameters. The decoding complexity of the proposed looped K-best decoder is analyzed and the gain from individual components/modules is quantified. Simulation results show the proposed MIMO-NOS scheme outperforms CRC-aided list decoding polar codes with maximum likelihood MIMO detection by 1 – 2 dB in various MIMO configurations for short (32 – 64 bits) message transmission.

References

[1] C. Bian, M. Yang, C.-W. Hsu, and H.-S. Kim, "Deep learning based near-orthogonal superposition code for short message transmission," in *Proc. IEEE Int. Conf. Commun.*, May 2022, pp. 3892–3897.

[2] Z. Dawy, W. Saad, A. Ghosh, J. G. Andrews, and E. Yaacoub, "Toward massive machine type cellular communications," *IEEE Wireless Commun.*, vol. 24, no. 1, pp. 120–128, Feb. 2016.

[3] C. Bockelmann et al., "Massive machine-type communications in 5G: Physical and MAC-layer solutions," *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 59–65, Sep. 2016.

[4] M. R. Palattella et al., "Internet of Things in the 5G era: Enablers, architecture, and business models," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 510–527, Mar. 2016.

[5] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.

[6] H. Gamage, N. Rajatheva, and M. Latva-Aho, "Channel coding for enhanced mobile broadband communication in 5G systems," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, Jun. 2017, pp. 1–6.

[7] M. C. Coşkun et al., "Efficient error-correcting codes in the short blocklength regime," *Phys. Commun.*, vol. 34, pp. 66–79, Jun. 2019.

[8] H.-S. Kim, "HDM: Hyper-dimensional modulation for robust low-power communications," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.

[9] C.-W. Hsu and H.-S. Kim, "Collision-tolerant narrowband communication using non-orthogonal modulation and multiple access," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.

[10] C.-W. Hsu and H.-S. Kim, "Non-orthogonal modulation for short packets in massive machine type communications," in *Proc. IEEE Global Commun. Conf.*, Dec. 2020, pp. 1–6.

[11] A. Joseph and A. R. Barron, "Fast sparse superposition codes have near exponential error probability for $R<C$," *IEEE Trans. Inf. Theory*, vol. 60, no. 2, pp. 919–942, Feb. 2014.

[12] T. Gruber, S. Cammerer, J. Hoydis, and S. T. Brink, "On deep learning-based channel decoding," in *Proc. 51st Annu. Conf. Inf. Sci. Syst. (CISS)*, Mar. 2017, pp. 1–6.

[13] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *Proc. 54th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2016, pp. 341–346.

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

[15] 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.

[16] Y. Jiang, H. Kim, H. Asnani, S. Kannan, S. Oh, and P. Viswanath, "Turbo autoencoder: Deep learning based channel codes for point-to-point communication channels," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 2754–2764.

[17] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," in *Proc. IEEE 18th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jul. 2017, pp. 1–5.

[18] T. Wang, L. Zhang, and S. C. Liew, "Deep learning for joint MIMO detection and channel decoding," in *Proc. IEEE 30th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Sep. 2019, pp. 1–7.

[19] C. Rush, A. Greig, and R. Venkataramanan, "Capacity-achieving sparse superposition codes via approximate message passing decoding," *IEEE Trans. Inf. Theory*, vol. 63, no. 3, pp. 1476–1500, Mar. 2017.

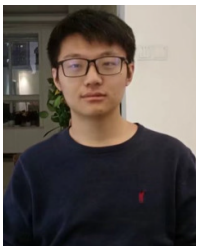
[20] A. Greig and R. Venkataramanan, "Techniques for improving the finite length performance of sparse superposition codes," *IEEE Trans. Commun.*, vol. 66, no. 3, pp. 905–917, Mar. 2018.

[21] B. Hassibi and B. M. Hochwald, "High-rate codes that are linear in space and time," *IEEE Trans. Inf. Theory*, vol. 48, no. 7, pp. 1804–1824, Jul. 2002.

[22] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Trans. Inf. Theory*, vol. 45, no. 5, pp. 1456–1467, Jul. 1999.

[23] M. Biguesh and A. B. Gershman, "Training-based MIMO channel estimation: A study of estimator tradeoffs and optimal training signals," *IEEE Trans. Signal Process.*, vol. 54, no. 3, pp. 884–893, Mar. 2006.

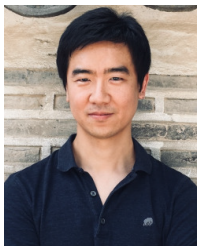
[24] Y. Li, "Simplified channel estimation for OFDM systems with multiple transmit antennas," *IEEE Trans. Wireless Commun.*, vol. 1, no. 1, pp. 67–75, Jan. 2002.
[25] M. K. Ozdemir and H. Arslan, "Channel estimation for wireless OFDM systems," *IEEE Commun. Surveys Tuts.*, vol. 9, no. 2, pp. 18–48, 2nd Quart., 2007.
[26] M. Yang, C. Bian, and H.-S. Kim, "OFDM-guided deep joint source channel coding for wireless multipath fading channels," *IEEE Trans. Cognit. Commun. Netw.*, vol. 8, no. 2, pp. 584–599, Jun. 2022.
[27] Z. Guo and P. Nilsson, "Algorithm and implementation of the K-best sphere decoding for MIMO detection," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 3, pp. 491–503, Mar. 2006.
[28] L. G. Barbero and J. S. Thompson, "Fixing the complexity of the sphere decoder for MIMO detection," *IEEE Trans. Wireless Commun.*, vol. 7, no. 6, pp. 2131–2142, Jun. 2008.
[29] S. Wiseman and A. M. Rush, "Sequence-to-sequence learning as beam-search optimization," 2016, *arXiv:1606.02960*.
[30] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Trans. Inf. Theory*, vol. 48, no. 8, pp. 2201–2214, Aug. 2002.
[31] *NR; Multiplexing and Channel Coding (Release 15)*, document GT 38.212, 3rd Generation Partnership Project, 2018.
[32] C. Martinez, "Partial quicksort," in *Proc. 6th ACMSIAM Workshop Algorithm Eng. Exp., 1st ACM-SIAM Workshop Analytic Algorithmics Combinatorics*, 2004, pp. 224–228.
[33] *Short Block-Length Design*, 3rd Generation Partnership Project, document GR1-1610141, 2016.



Chenghong Bian received the B.S. degree from the Physics Department, Tsinghua University, in 2020, and the M.S. degree from the EECS Department, University of Michigan, in 2022. He is currently pursuing the Ph.D. degree with the Department of Electrical and Electronic Engineering, Imperial College London. His research interests include semantic communications, channel coding, and deep learning.

Chin-Wei Hsu received the B.S. degree in electrical engineering and the M.S. degree in communication engineering from National Taiwan University, Taipei, Taiwan, in 2015 and 2017, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Michigan, Ann Arbor, MI, USA, in 2023. His research interests include novel modulation, coding schemes, and low-power design for wireless communications.

Changwoo Lee received the B.S. and M.S. degrees in electronic engineering from Hanyang University, Seoul, South Korea. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Michigan, Ann Arbor, MI, USA. His research interests include efficient deep learning and inverse problems.



Hun-Seok Kim (Senior Member, IEEE) received the B.S. degree in electrical engineering from Seoul National University, Seoul, South Korea, in 2001, and the Ph.D. degree in electrical engineering from the University of California at Los Angeles, Los Angeles, CA, USA, in 2010. He is currently an Associate Professor with the University of Michigan, Ann Arbor, MI, USA. His research interests include system analysis, novel algorithms, and VLSI architectures for low-power/high-performance wireless communications, signal processing, computer vision, and machine learning systems. He was a recipient of the DARPA Young Faculty Award in 2018 and the NSF CAREER Award in 2019. He is an Associate Editor of IEEE TRANSACTIONS ON MOBILE COMPUTING.