

Detect to Learn: Structure Learning With Attention and Decision Feedback for MIMO-OFDM Receive Processing

Jiarui Xu^{ID}, *Graduate Student Member, IEEE*, Lianjun Li^{ID}, Lihong Zheng, *Fellow, IEEE*, and Lingjia Liu^{ID}, *Senior Member, IEEE*

Abstract—The limited over-the-air (OTA) pilot symbols in multiple-input-multiple-output orthogonal-frequency-division-multiplexing (MIMO-OFDM) systems presents a major challenge for detecting transmitted data symbols at the receiver, especially for machine learning-based approaches. While it is crucial to explore effective ways to exploit pilots, one can also take advantage of the data symbols to improve detection performance. Thus, this paper introduces an online attention-based approach, namely RC-AttStructNet-DF, that can efficiently utilize pilot symbols and be dynamically updated with the detected payload data using the decision feedback (DF) mechanism. Reservoir computing (RC) is employed in the time domain network to facilitate efficient online training. The frequency domain network adopts the novel 2D multi-head attention (MHA) module to capture the time and frequency correlations, and the structural-based StructNet to facilitate the DF mechanism. The attention loss is designed to learn the frequency domain network. The DF mechanism further enhances detection performance by dynamically tracking the channel changes through detected data symbols. The effectiveness of the RC-AttStructNet-DF approach is demonstrated through extensive experiments in MIMO-OFDM and massive MIMO-OFDM systems with different modulation orders and under various scenarios.

Index Terms—Symbol detection, decision feedback, attention mechanism, multi-head attention, MIMO-OFDM, massive MIMO-OFDM, online training, structural knowledge.

I. INTRODUCTION

DEEP learning (DL) has attracted significant attention due to its overwhelming privilege in computer vision (CV), natural language processing (NLP), and robotics. DL has also demonstrated great potential in MIMO-OFDM symbol detection tasks. Efforts have been devoted to utilizing multi-layer

perceptron (MLP) [1], convolution neural network (CNN) [2], [3], [4], long short-term memory (LSTM) network [5], [6], or generative adversarial network (GAN) [7] to jointly estimate the channel and detect symbols. These approaches do not rely on explicit modeling of the system and the estimation of the channel state information (CSI), but instead, they treat the model as a black box and use neural networks (NNs) to deal with more complicated wireless systems with non-linear device components. Moreover, DL-based approaches offer a new avenue for symbol detection tasks with a large number of OFDM subcarriers and high modulation orders, thanks to their lower computational complexity compared to conventional optimal detection algorithms [8].

While DL-based approaches are promising, such approaches have their own challenges to be applied in real-world MIMO-OFDM systems. One of the main challenges is the data requirement. Training NNs requires a huge volume of labeled data along with a long training time. In modern cellular networks, OTA labeled data is scarce and extremely expensive to obtain, resulting in limited training data for symbol detection tasks. Attempts have been made to address such issues by training with extensive channel realizations offline and initializing the network with offline weights for online adaptation [1]. However, such approaches are hard to be adopted to the dynamically changing environments in practice, where the systems have different transmission modes with link adaptation, rank adaptation, and schedules operating on a subframe (1 millisecond) basis [8], [9]. The discrepancy between modes for offline and online training can result in model mismatch and prohibit the offline weights from being utilized online.

Another challenge is how to design an interpretable network with domain knowledge. Learning both the known and unknown features requires a large amount of training data. The desired network should embed what we have known into the architecture instead of learning everything from scratch. Recent advances employ deep-unfolding NNs to improve network interoperability for detection [10], [11], [12], [13], transceiver design [14], [15], and channel estimation [16]. Specifically, for symbol detection, such approaches [11], [12], [13] build the network by unfolding conventional iterative algorithms and thus reduce the number of trainable parameters in the network. For

Manuscript received 4 November 2022; revised 28 February 2023 and 17 May 2023; accepted 23 June 2023. Date of publication 3 August 2023; date of current version 17 January 2024. The work is supported by US National Science Foundation (NSF) and Intel under grants CNS-2003059, CNS-2002908 and CCF-1937487. The associate editor coordinating the review of this article and approving it for publication was M. Vaezi. (*Corresponding author: Lingjia Liu.*)

Jiarui Xu and Lingjia Liu are with the Wireless@Virginia Tech, Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061 USA (e-mail: ljliu@ieee.org).

Lianjun Li is with Samsung Research America, Plano, TX 75023 USA.

Lihong Zheng is with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139 USA.

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TCOMM.2023.3292468>.

Digital Object Identifier 10.1109/TCOMM.2023.3292468

0090-6778 © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See <https://www.ieee.org/publications/rights/index.html> for more information.

example, MMNet [12] converts each iteration of the iterative soft-thresholding algorithms into network layers and only learns a few parameters of the algorithm with NN in each layer. It has shown superior performance in independent and identically distributed (i.i.d.) Gaussian channels and 3D MIMO channels released by 3rd Generation Partnership Project (3GPP) [17]. However, due to the stack of multiple layers, the number of parameters grows, leading to a demand for a large-size training dataset. Furthermore, the assumption of the perfect CSI knowledge also limits its utilization in practice.

Our previous RC based approaches [18], [19], [20] adopts the RC in the time domain to conduct symbol detection and thus can be learned with only a limited amount of online training data. More recently, RC-Struct [21] extends the idea of the time domain RC-based approaches by introducing an extra frequency domain structure-based network. The network consists of a time domain RC and a frequency domain classifier. By processing in both domains, the time domain convolution and superposition operation of the wireless channel, the time-frequency structure of OFDM, and the frequency domain structure of the repetitive modulation constellation pattern, can all be embedded into the underlying NNs to take advantage of the available structural knowledge of the MIMO-OFDM system. The incorporation of such structural knowledge helps to significantly reduce the needed training overhead and improve the system performance in a realistic environment. However, these RC-based methods only learn from the pilot OFDM symbols and then test on the rest data OFDM symbols. While they still work in the scenario where the channel is gradually evolving over different OFDM symbols within a subframe, it does not learn the dynamic features that are provided by the data symbols.

In this paper, we introduce an online attention-based approach, RC-AttStructNet-DF, to tackle the above challenges. We follow our previous work, RC-Struct, to embed structural knowledge of the MIMO-OFDM system into the network architecture. To further improve detection performance, we adopt an additional attention mechanism in our approach, which is achieved by a 2D MHA module and attention loss. Inspired by the MHA module in the Transformer [22], we develop the 2D MHA to capture time and frequency correlations in a two-dimensional manner. The attention loss is designed for the frequency domain network to assign different weights to the training loss of different training samples based on their confidence levels. Moreover, unlike RC-Struct which only learns from the pilot symbols, RC-AttStructNet-DF also takes advantage of the data symbols. Specifically, a DF mechanism, which takes the detected data symbols as the training data, is used to dynamically track the channel variations within a subframe. To mitigate the error propagation of the DF procedure, the frequency domain StructNet is designed on top of the RC-Struct, where a parameter estimation (PE) layer is additionally adopted. The customized StructNet is shown to be robust to incorrect labels owing to the embedded structural information, making it a suitable network to be applied with the DF process. The introduced PE layer further facilitates the DF mechanism by learning the network parameters with the underlying NN. Evaluation

results demonstrate the effectiveness of adopting the attention mechanism to improve detection performance, and the benefits of employing the StructNet along with the DF mechanism to mitigate the issue of error propagation and dynamically update network parameters. Meanwhile, extensive experiments have been conducted to show the considerable performance gain of RC-AttStructNet-DF over the conventional model-based methods and the state-of-the-art learning-based approaches in both the MIMO-OFDM system and the massive MIMO-OFDM system under the 3GPP 3D channel [17]. Evaluation results also demonstrate that RC-AttStructNet-DF can perform better than the conventional approaches with a practical pilot pattern specified by the 3GPP 5G NR. Our contributions are summarized as the following:

- We develop an attention-based symbol detection approach, where attention is achieved by adopting a novel 2D MHA module and attention loss. The 2D MHA is constructed based on the MHA module in the Transformer, which captures time and frequency correlations in a two-dimensional manner. The attention loss is designed to assign different weights to the training loss of different training samples according to their confidence levels.
- We exploit the DF mechanism to take advantage of the data symbols to improve detection performance. Specifically, the DF algorithm updates the network weights by utilizing the detected data symbols as the training data so that the channel changes across different OFDM data symbols can be dynamically tracked.
- We design a learning-efficient network, StructNet, in the frequency domain on top of the existing RC-Struct. The customized StructNet is shown to be robust to incorrect labels owing to the embedded structural information, which effectively mitigates the issue of error propagation typically encountered in the DF procedure. Moreover, the introduced PE layer in StructNet enables the network to dynamically update its parameters, further facilitating the underlying DF mechanism.
- Extensive performance evaluations have been conducted to show that RC-AttStructNet-DF outperforms RC-Struct in the MIMO-OFDM system under various scenarios with relatively high user mobility. Furthermore, RC-AttStructNet-DF demonstrates its effectiveness in large-scale antenna array settings, extending the application scenario from the MIMO system to the massive MIMO system. Experimental results also reveal the potential of employing RC-AttStructNet-DF in a practical pilot pattern specified by the 3GPP 5G NR.

The rest of the paper is organized as follows. Sec. II presents the MIMO-OFDM system and problem statement. Sec. III briefly discusses RC, the MHA module, and the atomic decision neuron network (ADNN) exploited in RC-Struct. Sec. IV introduces the frequency-domain network, StructNet, in a simple MIMO system. Then Sec. V discusses how to apply StructNet in the MIMO-OFDM system and introduces the RC-AttStructNet-DF. Complexity analysis is provided in Sec. VI. A toy experiment for analyzing StructNet

is shown in Sec. VII. Sec. VIII compares the performance of RC-AttStructNet-DF with state-of-the-art methods in both the MIMO-OFDM system and the massive MIMO-OFDM system. The paper is concluded in Sec. IX.

Notations: Scalar, vector, and matrix are denoted by non-bold letter, bold lowercase letter, and bold uppercase letter. The superscripts $(\cdot)^t$ and $(\cdot)^f$ are used to differentiate the time domain and the frequency domain signals, respectively. $\mathbb{C}(\mathbb{R})$ represents the complex (real) number set. Other sets are denoted by calligraphic letters.

II. PROBLEM FORMULATION

We consider a MIMO-OFDM system with N_t transmit antennas, N_r receive antennas, and N_{sc} subcarriers. The information is modulated in the frequency domain on a subframe basis, where each subframe consists of N OFDM symbols. The n -th OFDM symbol in the frequency domain can be represented as $\mathbf{X}_n^f \in \mathbb{C}^{N_t \times N_{sc}}$. The frequency domain OFDM symbols are converted to the time domain through an inverse fast Fourier transform (IFFT) and cyclic prefix (CP) addition with length N_{cp} . Then all the time domain OFDM symbols are concatenated together to form the transmitted time domain signal $\mathbf{X}^t = [\mathbf{X}_0^t, \mathbf{X}_1^t, \dots, \mathbf{X}_{N-1}^t] \in \mathbb{C}^{N_t \times N(N_{sc} + N_{cp})}$, where $\mathbf{X}_n^t \in \mathbb{C}^{N_t \times (N_{sc} + N_{cp})}$ is the n -th OFDM symbol in the time domain.

Denote the time domain signal at the n_t -th transmit antenna as $\mathbf{x}_{n_t}^t \in \mathbb{C}^{N(N_{sc} + N_{cp})}$, and the L -tap channel between the receive antenna n_r and transmit antenna n_t as $\mathbf{h}_{n_r, n_t}^t \in \mathbb{C}^L$, where the channel is gradually evolving across different OFDM symbols. Then at the receiver side, the received signal can be written as

$$\mathbf{y}_{n_r}^t = \sum_{n_t=0}^{N_t-1} \mathbf{h}_{n_r, n_t}^t \otimes g(\mathbf{x}_{n_t}^t) + \mathbf{w}^t, \quad (1)$$

where $\mathbf{y}_{n_r}^t \in \mathbb{C}^{N(N_{sc} + N_{cp})}$ is the received signal at the n_r -th receive antenna; $\otimes$ stands for the convolution operation; $g(\cdot)$ represents the non-linear distortion caused by transmitter components, such as power amplifier (PA); $\mathbf{w}^t$ denotes the additive white Gaussian noise (AWGN). Let $\mathbf{Y}^t \in \mathbb{C}^{N_r \times N(N_{sc} + N_{cp})}$ represent the received time domain signal for all the receive antennas, and $\mathbf{Y}_n^t \in \mathbb{C}^{N_r \times (N_{sc} + N_{cp})}$ is the n -th received OFDM symbol in the time domain. The frequency domain counterpart of the received signal $\mathbf{Y}_n^f \in \mathbb{C}^{N_r \times N_{sc}}$ is acquired by removing CP and following with a fast Fourier transform (FFT). The notations are summarized in Table I.

The MIMO-OFDM symbol detection task is to recover the transmitted symbol $\mathbf{X}_n^f$ from the received time-domain observations $\mathbf{Y}_n^t$ for each OFDM symbol. Pilot symbols, which are known at both the transmitter and receiver sides, will be embedded in each subframe to facilitate the detection of unknown data symbols. For ease of discussion, we assume the first N_p OFDM symbols are the pilots, and the rest of the N_d symbols are the unknown data. However, the introduced approach can also be applied to other pilot patterns, which will be shown in Sec. VIII-E. In this work, the training dataset for our NN-based approach consists of the transmitted and received time domain pilot signals $\mathbf{X}_n^t$ and $\mathbf{Y}_n^t$ along with the frequency domain pilot symbols $\mathbf{X}_n^f$.

TABLE I
NOTATION IN THE SYSTEM

Symbol	Definition
N_t	Number of transmit antennas
N_r	Number of receive antennas
N_{sc}	Number of OFDM subcarriers
N_{cp}	Length of cyclic prefix
N_p	Number of pilot symbols in one OFDM frame (training set)
N_d	Number of data symbols in one OFDM frame (testing set)
N	Total number of symbols in one OFDM frames
L	Total number of channel delays
$\mathbf{Y}^t \in \mathbb{C}^{N_r \times N(N_{sc} + N_{cp})}$	The received time domain signal
$\mathbf{X}^t \in \mathbb{C}^{N_t \times N(N_{sc} + N_{cp})}$	The transmitted time domain signal
$\mathbf{h}_{n_r, n_t}^t \in \mathbb{C}^L$	The channel impulse response between receiver n_r and transmitter n_t
$\mathbf{y}_{n_r}^t \in \mathbb{C}^{N(N_{sc} + N_{cp})}$	The received signal at n_r -th receiver in time domain
$\mathbf{x}_{n_t}^t \in \mathbb{C}^{N(N_{sc} + N_{cp})}$	The transmitted signal at n_t -th transmitter in time domain
$\mathbf{Y}_n^t \in \mathbb{C}^{N_r \times (N_{sc} + N_{cp})}$	The received n -th OFDM symbols in time domain
$\mathbf{X}_n^t \in \mathbb{C}^{N_t \times (N_{sc} + N_{cp})}$	The transmitted n -th OFDM symbols in time domain
$\mathbf{Y}_n^f \in \mathbb{C}^{N_r \times N_{sc}}$	The received n -th OFDM symbols in frequency domain
$\mathbf{X}_n^f \in \mathbb{C}^{N_t \times N_{sc}}$	The transmitted n -th OFDM symbols in frequency domain

III. PRELIMINARY

A. Reservoir Computing

RC is a recurrent neural network (RNN) based approach that can process sequential data. It only conducts training on the output layer and fixes the randomly initialized weights of the input layer and the RNN-based reservoir. As the detailed discussion about RC has been provided in our previous work [18], [19], [20], [21], [23], [24], we mainly focus on the processing procedures of RC. Denote the input data as $\mathbf{U} \in \mathbb{C}^{N_{in} \times T}$, where $\mathbf{u}(m)$ represents the m -th column of $\mathbf{U}$. The processing of RC at the m -th time step can be represented by the following state transition equation and the output equation:

$$\mathbf{s}(m) = f(\mathbf{W}_{in} \mathbf{s}(m-1) + \mathbf{W}_{in} \mathbf{u}(m)), \quad \hat{\mathbf{o}}(m) = \mathbf{W}_{out} \mathbf{z}(m), \quad (2)$$

where $\mathbf{s}(m) \in \mathbb{C}^{N_n}$ is the state vector; $\mathbf{W}_{in} \in \mathbb{C}^{N_n \times N_{in}}$ and $\mathbf{W}_{out} \in \mathbb{C}^{N_{out} \times N_n}$ are the fixed input weight matrix and reservoir transition matrix, respectively; the $f(\cdot)$ is the hyperbolic tangent function; $\hat{\mathbf{o}}(m) \in \mathbb{C}^{N_{out}}$ is the predicted output at m -th time step; $\mathbf{z}(m) = [\mathbf{s}(m)^T, \mathbf{u}(m)^T]^T \in \mathbb{C}^{N_n + N_{in}}$ is the state and input concatenation; $\mathbf{W}_{out} \in \mathbb{C}^{N_{out} \times (N_n + N_{in})}$ is the trainable output weight matrix. During training, the output weight matrix is learned by the least square (LS) solution

$$\hat{\mathbf{W}}_{out} = \arg \min_{\mathbf{W}_{out}} \|\mathbf{W}_{out} \mathbf{Z} - \mathbf{O}\|_F^2 = \mathbf{O} \mathbf{Z}^\dagger, \quad (3)$$

where $\mathbf{Z} = [\mathbf{z}(0), \mathbf{z}(1), \dots, \mathbf{z}(T-1)] \in \mathbb{C}^{(N_n + N_{in}) \times T}$ is the concatenation of $\mathbf{z}(m)$, $\mathbf{O} \in \mathbb{C}^{N_{out} \times T}$ is the training label, and $\mathbf{Z}^\dagger$ is the Moore-Penrose inverse of $\mathbf{Z}$.

B. Multi-Head Attention

The MHA module is first introduced in the Transformer [22], which allows the model to attend to information from different representation subspaces. As shown in Fig. 1, the MHA module consists of multiple attention blocks. For each block, the attention function maps the input into the query, key, and value subspaces as the query, key, and value embeddings. The output of the attention block, or the attention head, is obtained by the scaled dot product of the three embeddings. Particularly, the attention block can be computed

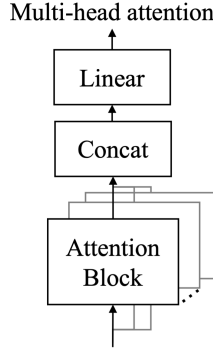


Fig. 1. The MHA module.

by the following equations:

$$\mathbf{Q} = \mathbf{U} \cdot \mathbf{W}_q, \mathbf{K} = \mathbf{U} \cdot \mathbf{W}_k, \mathbf{V} = \mathbf{U} \cdot \mathbf{W}_v, \quad (4)$$

$$\text{head} = f_{sm}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{N_k}}\right) \cdot \mathbf{V}, \quad (5)$$

where $\mathbf{U} \in \mathbb{R}^{N_{in1} \times N_{in2}}$ is the input to the attention block; $\mathbf{W}_q \in \mathbb{R}^{N_{in2} \times N_k}$, $\mathbf{W}_k \in \mathbb{R}^{N_{in2} \times N_k}$, and $\mathbf{W}_v \in \mathbb{R}^{N_{in2} \times N_v}$ are linear projections that project the input to the query, key, and value embedding spaces, respectively; $f_{sm}(\cdot)$ represents the softmax function; N_k and N_v are the dimensions of key embedding and value embedding, respectively. In the MHA module, the heads of each attention block are aggregated by concatenation and a linear projection:

$$\text{MultiHead} = \text{Concat}(\text{head}_0, \text{head}_1, \dots, \text{head}_{N_a-1}) \cdot \mathbf{W}_o, \quad (6)$$

where $\mathbf{W}_o \in \mathbb{R}^{N_a N_v \times N_o}$ is the output linear project, N_a is the number of attention blocks, and N_o is the output dimension.

C. Atomic Decision Neuron Network

The ADNN [21], [25] is a frequency domain symbol detection network that utilizes a single binary classifier to conduct the multi-class detection task. The design of the network exploits the repetitive structure of the modulation constellation, which improves the training efficiency by reducing the network size and utilizing training samples more efficiently.

1) *Problem Formulation*: Consider a $N_r \times N_t$ MIMO system in the frequency domain,

$$\mathbf{y}^f = \mathbf{H}^f \mathbf{x}^f + \mathbf{w}^f, \quad (7)$$

where $\mathbf{y}^f \in \mathbb{C}^{N_r}$ is the received signal, $\mathbf{H}^f \in \mathbb{C}^{N_r \times N_t}$ represents the channel in the frequency domain, $\mathbf{x}^f \in \mathbb{C}^{N_t}$ stands for the transmitted M -quadrature amplitude modulation (M -QAM) symbol, and $\mathbf{w}^f \in \mathbb{C}^{N_r}$ is the additive noise that may not need to be Gaussian noise. The real-valued form of $\mathbf{y}^f$ and $\mathbf{x}^f$, which are used as the input and output of the network, are obtained by $\tilde{\mathbf{y}}^f = f_R(\mathbf{y}^f)$ and $\tilde{\mathbf{x}}^f = f_R(\mathbf{x}^f) \in \mathcal{A}^{2N_t}$, where $f_R(\cdot)$ is the complex to real value function $f_R(\mathbf{x}) = [\Re\{\mathbf{x}\}^T, \Im\{\mathbf{x}\}^T]^T$ and $\mathcal{A}$ is the set $\{-2K-1, -2K+1, \dots, 2K-1, 2K+1\}$ with $K = \frac{\sqrt{M}-2}{2}$. Note that the $\tilde{\mathbf{x}}^f$

are now the $\sqrt{M}$ pulse amplitude modulation ($\sqrt{M}$ -PAM) symbols. The real-valued form of channel is acquired by:

$$\tilde{\mathbf{H}}^f = \begin{bmatrix} \Re\{\mathbf{H}^f\}, -\Im\{\mathbf{H}^f\} \\ \Im\{\mathbf{H}^f\}, \Re\{\mathbf{H}^f\} \end{bmatrix}. \quad (8)$$

Then the problem becomes a classification task with labels in the set $\mathcal{A}$, which can be approximated by the Naive Bayesian approximation

$$\arg\max_{\tilde{\mathbf{x}}^f} P\{\tilde{\mathbf{x}}^f | \tilde{\mathbf{y}}^f\} \approx \arg\max_{\tilde{\mathbf{x}}^f} \prod_{n \in \{0, 1, \dots, 2N_t-1\}} P_n\{\tilde{x}_n^f | \tilde{\mathbf{y}}^f\}, \quad (9)$$

where the $P_n\{\cdot | \tilde{\mathbf{y}}^f\}$ represents the marginal distribution of the n -th entry of $\tilde{\mathbf{x}}^f$. The ADNN is designed to learn functions that approximate the $P_n\{\cdot | \tilde{\mathbf{y}}^f\}$.

2) *Binary Detection*: For simplicity, we start with the QPSK detection, where the $\tilde{x}_n^f$ is in the class set $\{+1, -1\}$. Denote the function approximated by the binary classifier as $\mathcal{B}_n$. We can directly apply the binary classifier to estimate the marginal likelihood ratio with

$$\frac{P_n\{\hat{x}_n^f = +1 | \tilde{\mathbf{y}}^f\}}{P_n\{\hat{x}_n^f = -1 | \tilde{\mathbf{y}}^f\}} \approx \frac{\mathcal{B}_n(\hat{b}_n = +1; \tilde{\mathbf{y}}^f)}{\mathcal{B}_n(\hat{b}_n = -1; \tilde{\mathbf{y}}^f)} =: \mathcal{L}_b^{+-}(\tilde{\mathbf{y}}^f), \quad (10)$$

where $\hat{x}_n^f$ is the estimated transmitted symbol, $\hat{b}_n$ is the predicted binary label, and $\mathcal{L}_b^{+-}$ denotes the likelihood ratio of the binary classifier.

3) *Multi-Class Detection*: When transmitting the M -QAM symbol, we perform the multi-class classification task with the class set $\mathcal{A}$ adopting the following shifting principle [25]:

$$\frac{P_n\{\hat{x}_n^f = -2k+1 | \tilde{\mathbf{y}}^f\}}{P_n\{\hat{x}_n^f = -2k-1 | \tilde{\mathbf{y}}^f\}} = \frac{P_n\{\hat{x}_n^f = +1 | \tilde{\mathbf{y}}^f + 2k \cdot \tilde{\mathbf{h}}_n^f\}}{P_n\{\hat{x}_n^f = -1 | \tilde{\mathbf{y}}^f + 2k \cdot \tilde{\mathbf{h}}_n^f\}}, \quad (11)$$

where $\tilde{\mathbf{h}}_n^f$ is the n -th column of the channel $\tilde{\mathbf{H}}^f$ and $k = -K, -K+1, \dots, +K$. The principle is based on the fact that the constellation points of the QAM modulation share the same distance 2 and have a repetitive structure. By shifting the received signal $\tilde{\mathbf{y}}^f$ in the direction of $\tilde{\mathbf{h}}_n^f$ with step size $2k$, which corresponds to $\tilde{\mathbf{y}}^f + 2k \cdot \tilde{\mathbf{h}}_n^f$, the transmit symbol $-2k+1$ and $-2k-1$ are shifted to $+1$ and -1 , respectively. Note that the binary label $+1$ and -1 is determined by the step size plus the transmit symbol. Then the likelihood ratio between class $-2k+1$ and $-2k-1$ can be estimated by conducting a binary classification in $\{+1, -1\}$ with input $\tilde{\mathbf{y}}^f + 2k \cdot \tilde{\mathbf{h}}_n^f$, which is denoted as

$$\frac{P_n\{\hat{x}_n^f = -2k+1 | \tilde{\mathbf{y}}^f\}}{P_n\{\hat{x}_n^f = -2k-1 | \tilde{\mathbf{y}}^f\}} \approx \mathcal{L}_b^{+-}(\tilde{\mathbf{y}}^f + 2k \cdot \tilde{\mathbf{h}}_n^f). \quad (12)$$

In this way, the multi-class classification is transformed into multiple binary detection processes. For ease of discussion, the operation $\tilde{\mathbf{y}}^f + 2k \cdot \tilde{\mathbf{h}}_n^f$ is referred to as the “shifting process” for the remainder of this paper, where the step size $2k$ is referred to as the “shifting parameter” and denoted as s_n . Note that while the above transformation is designed for QAM modulation, the same idea can be generalized to other modulation schemes with a customized shifting process.

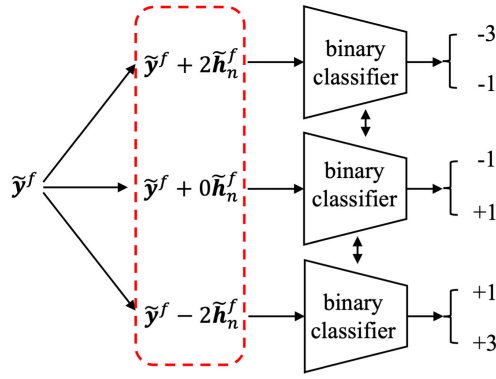


Fig. 2. The architecture of ADNN. “Share” means that the weights of all the binary classifiers are shared.

At the testing time, the received signal is tested along with all the possible values of the shifting parameter s_n in set $\mathcal{S} = \{-2K, -2K + 2, \dots, 2K - 2, 2K\}$. The probability of each class can be obtained by collecting all the likelihood ratios in eq.(12) with

$$P_n\{\hat{x}_n^f = -2k + 1|\tilde{\mathbf{y}}^f\} \approx P_n\{\hat{x}_n^f = -2K - 1|\tilde{\mathbf{y}}^f\} \prod_{k'=k}^K \mathcal{L}_b^{+-}(\tilde{\mathbf{y}}^f + 2k' \cdot \tilde{\mathbf{h}}_n^f), \quad (13)$$

where each probability can be computed by solving the equation $\sum_{a \in \mathcal{A}} P_n\{\hat{x}_n^f = a|\tilde{\mathbf{y}}^f\} = 1$. The estimated $\hat{x}_n^f$ is determined by the class with the maximum probability. In Fig. 2, we show the architecture of ADNN and an example of when it is tested in the 16-QAM case. The ADNN is employed as the frequency domain network in RC-Struct, where the channel $\tilde{\mathbf{h}}_n^f$ is estimated by the linear minimum mean square error (LMMSE) method.

4) *Construction of Binary Training Samples*: The complex-valued M -QAM symbols are converted to real-valued $\sqrt{M}$ -PAM symbols for training. For clarity, we define two types of training samples: the PAM training samples and the binary training samples. The PAM training samples refer to the real-valued transmit and receive signal pairs. The binary training samples are generated to train the designed binary classifier using the PAM training samples. For each PAM training sample, we construct two binary training samples, which include a positive binary sample and a negative binary sample. As mentioned in Sec. III-C.3, the binary label is determined by $b_n = s_n + \tilde{x}_n^f$. At the training time, the shifting parameter is set as $s_n = -\tilde{x}_n^f + b_n$ to control if a binary sample is positive or negative. Specifically, for each PAM training sample $(\tilde{x}_n^f, \tilde{\mathbf{y}}^f)$, positive and negative binary training samples are generated as

$$\begin{aligned} (b_n^+ = +1, \tilde{\mathbf{y}}^f, s_n^+ = -\tilde{x}_n^f + 1), \\ (b_n^- = -1, \tilde{\mathbf{y}}^f, s_n^- = -\tilde{x}_n^f - 1). \end{aligned} \quad (14)$$

In this way, each PAM training sample is augmented into two binary training samples and thus can be utilized more efficiently. In Tab. II, we show examples of positive and negative shifting parameters corresponding to 4-PAM symbols.

TABLE II
CONSTRUCTION OF BINARY TRAINING SAMPLE

Transmit symbol $\tilde{x}_n^f$	$s_n^+ = -\tilde{x}_n^f + 1$	$s_n^- = -\tilde{x}_n^f - 1$
-3	4	2
-1	2	0
+1	0	-2
+3	-2	-4

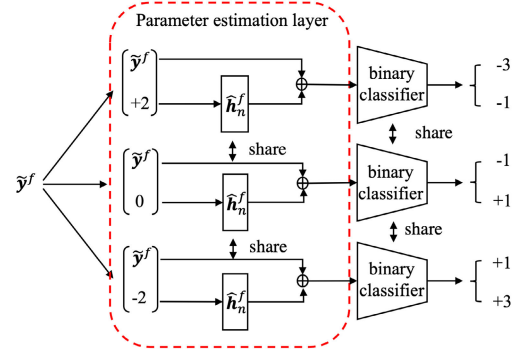


Fig. 3. The architecture of StructNet.

IV. FREQUENCY DOMAIN NETWORK — STRUCTNET

In this section, we introduce the frequency domain network, StructNet, in a MIMO system and analyze the properties of the network. The discussion of how to apply StructNet in the MIMO-OFDM symbol detection task is provided in Sec. V.

A. Design of StructNet

StructNet is a frequency domain network that builds upon ADNN, where it also embeds the repetitive structure of the modulation constellation into the network. In ADNN, the shifting process is performed using the ground truth channel. When adopted in RC-Struct, the LMMSE estimated channel is utilized due to the difficulty in obtaining perfect channel knowledge in practice. While the estimated channel can serve as an alternative to conduct the shifting process, it lacks the ability to adapt to changes in the environment. To address this issue, an additional PE layer is introduced in StructNet to estimate channel and perform the shifting process. The weights of the PE layer are dynamically updated according to channel variations without relying on perfect channel knowledge.

The architecture of StructNet is depicted in Fig. 3, comprising a PE layer and a binary classifier. The weights of the PE layer, denoted by $\hat{\mathbf{h}}_n^f$, also represent an estimate of the ground truth channel $\tilde{\mathbf{h}}_n^f$. The PE layer takes the shifting parameter $s_n \in \mathcal{S}$ as input and is implemented using a linear NN layer. The binary classifier is an MLP. The PE layer is initialized with the LMMSE estimated channel and updated along with the binary classifier through backpropagation. The network is trained using the cross-entropy loss.

One notable aspect of StructNet is that its PE layer for channel estimation is learned without assuming knowledge of the ground truth channel. Unlike other NN-based channel estimation approaches, which require ground truth channels to calculate the network loss, StructNet adopts a more realistic

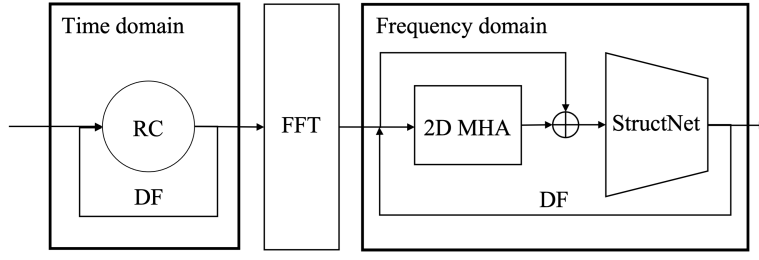


Fig. 4. Architecture of RC-AttStructNet-DF network.

setting and does not assume perfect CSI. Instead, the PE layer is trained based on the loss of the binary classification. Such an update of the PE layer is achieved by embedding the repetitive modulation constellation structure into the network design. In addition, the embedded structural information also enables StructNet to utilize training samples more efficiently, as discussed in Sec. III-C.4, and to be more robust to incorrect labels, as analyzed in Sec. IV-B. The robustness to incorrect labels makes StructNet less susceptible to error propagation when applied in the DF mechanism. Furthermore, the introduced PE layer further facilitates the DF approach due to the dynamic updating of the network parameters.

B. Training With Incorrect Labels

Due to the unique training sample construction process discussed in Sec. III-C.4, given any percentage of the incorrect PAM labels, there are at most 50% incorrect binary labels during the training stage. We first explain the reason for this conclusion with a concrete example and then discuss it in a more general case.

For ease of discussion, we use 4-PAM as an example, but the conclusion generalizes to $\sqrt{M}$ -PAM labels. Note that there are two kinds of labels in our setting: the PAM labels and the binary labels. The PAM label is the transmitted $\sqrt{M}$ -PAM symbol in the set $\mathcal{A}$. The binary label is in the set $\{+1, -1\}$, indicating whether a sample is positive or negative. Assume we have a correct PAM label with $\tilde{x}_n^f = -1$ and the corresponding incorrect PAM label is $\tilde{x}_n^f = +1$. Then the training samples created by the incorrect label $\tilde{x}_n^f = +1$ can be written as

$$\begin{aligned} (\bar{b}_n^+ = +1, \tilde{\mathbf{y}}^f, s_n^+ = -\tilde{x}_n^f + 1 = 0), \\ (\bar{b}_n^- = -1, \tilde{\mathbf{y}}^f, s_n^- = -\tilde{x}_n^f - 1 = -2). \end{aligned} \quad (15)$$

Note that the actual binary label is determined by $b_n = s_n + \tilde{x}_n^f$. As the actual PAM label is $\tilde{x}_n^f = -1$, the actual binary label for the constructed positive sample, however, is $b_n^+ = s_n^+ + \tilde{x}_n^f = -1 < 0$, resulting in an incorrect binary positive sample. For the negative sample, the actual binary label is $b_n^- = s_n^- + \tilde{x}_n^f = -3 < 0$, which is a correct binary sample.

In general, suppose $\tilde{x}_n^f$ is the incorrect PAM label corresponding to the actual PAM label $\tilde{x}_n^f$. The positive sample is created by setting $s_n^+ = -\tilde{x}_n^f + 1$ and the negative sample is constructed by setting $s_n^- = -\tilde{x}_n^f - 1$. As the actual PAM label is $\tilde{x}_n^f$, the actual binary label for the positive sample is $b_n^+ = \tilde{x}_n^f + s_n^+ = \tilde{x}_n^f - \tilde{x}_n^f + 1 = 1$ and the actual label for the negative sample is $b_n^- = \tilde{x}_n^f + s_n^- = \tilde{x}_n^f - \tilde{x}_n^f - 1 = -1$. The positive sample is considered to be incorrect if $b_n^+ = \tilde{x}_n^f - \tilde{x}_n^f + 1 < 0$,

i.e., $\tilde{x}_n^f - \tilde{x}_n^f < -1$. The negative sample is incorrect when $b_n^- = \tilde{x}_n^f - \tilde{x}_n^f - 1 > 0$, i.e., $\tilde{x}_n^f - \tilde{x}_n^f > 1$. Since $\tilde{x}_n^f, \tilde{x}_n^f \in \mathcal{A}$ and $\tilde{x}_n^f \neq \tilde{x}_n^f$, the distance $\tilde{x}_n^f - \tilde{x}_n^f$ can at most satisfy one of the inequalities between $\tilde{x}_n^f - \tilde{x}_n^f < -1$ and $\tilde{x}_n^f - \tilde{x}_n^f > 1$ for any value of $\tilde{x}_n^f$ and $\tilde{x}_n^f$. Thus, there is at least one correct positive sample or correct negative sample for any incorrect PAM sample. The percentage of incorrect binary samples is at most 50%. The incorrect percentage only equals 50% when all the training PAM labels are incorrect. This unique property of StructNet makes it less affected by detection errors. As one of the major issues of DF-based approaches is error propagation, StructNet can mitigate the issue by showing robustness to the detection errors and therefore work well with the DF approach.

V. INTRODUCED APPROACH — RC-ATTSTRUCTNET-DF

In this section, we introduce the RC-AttStructNet-DF approach. As shown in Fig. 4, the architecture of RC-AttStructNet-DF consists of three essential components: the time domain RC network, the 2D MHA module, and the frequency domain StructNet network. The time domain RC network is exploited to decouple the transmitted data streams and deconvolve the channel for equalization [19], [20]. Furthermore, we design a DF mechanism to dynamically update the network with the detected data symbols, especially in high mobility scenarios. The RLS algorithm is used in the time domain as part of the DF mechanism to adaptively update the weights on an OFDM symbol basis. In the frequency domain, the 2D MHA module is designed to capture the time and frequency correlation of the signal. Subsequently, the frequency domain network, StructNet, is utilized to conduct the multi-class classification. The residual connection [26] is employed to connect the 2D MHA and the StructNet. Additionally, we develop an attention loss for the DF mechanism in the frequency domain, which weights the training loss of different samples based on their confidence level. In the following subsections, we provide detailed training and testing procedures for our approach.

A. RC With DF

In the time domain, the convolution and the superposition operation of the wireless channel are conducted on the transmit signal. The RC reverses such an operation by jointly decoupling the different transmitted data streams and deconvolving the channel for equalization [19], [20].

1) *Learning From Pilot Symbols*: The input to the RC network is the time domain received signal $\mathbf{Y}^t$ and the target output is the time domain transmitted signal $\mathbf{X}^t$. The training dataset for learning from pilot symbols can be represented as

$$\mathcal{D}_{rc} \triangleq \{\mathbf{Y}_n^t, \mathbf{X}_n^t\}_{n=0}^{N_p-1}. \quad (16)$$

The initial output weights of RC are learned with the pilot symbols through the LS solution shown in eq. (3).

2) *DF With Data Symbols*: The DF mechanism is utilized to dynamically update the output weights of RC with the detected data symbols in a symbol-by-symbol manner. During the DF procedure, the RLS [27], [28] method is adopted to recursively update the output weights. The initial weights for the RLS algorithm are the weights learned from the pilot symbols.

Specifically, we use RC trained by the pilot symbols to generate the estimation for the first data symbol $\hat{\mathbf{X}}_{N_p}^t \in \mathbb{C}^{N_t \times (N_{cp} + N_{sc})}$. For the n th data symbol, the estimation $\hat{\mathbf{X}}_n^t$ ($n = N_p + 1, \dots, N - 1$) is detected by the RC learned with the $(n - 1)$ -th data symbol. The estimation is then converted into frequency domain $\hat{\mathbf{X}}_n^f$ and then each symbol is mapped to the nearest constellation points, which generates $\bar{\mathbf{X}}_n^f$. The frequency domain $\bar{\mathbf{X}}_n^f$ is transformed back to the time domain $\bar{\mathbf{X}}_n^t$ through the IFFT operation. Then the output weights of RC are recursively updated by minimizing the objective:

$$\arg \min_{\mathbf{W}_{out}^{(n)}(m)} \sum_{m'=0}^m \alpha^{m-m'} \|\mathbf{W}_{out}^{(n)}(m') \mathbf{z}_n(m') - \bar{\mathbf{x}}_n^t(m')\|_2^2, \quad (17)$$

where $\mathbf{W}_{out}^{(n)}(m)$ is the weight learned by the n -th data symbol at step m ; $\mathbf{z}_n(m) \in \mathbb{C}^{N_n + N_t}$ is the concatenation of the RC state and input for the n -th data symbol at step m ; $\bar{\mathbf{x}}_n^t(m) \in \mathbb{C}^{N_t}$ is the m -th column of $\bar{\mathbf{X}}_n^t$; $\alpha \in (0, 1]$ is the forgetting factor; and $m = 0, 1, \dots, N_{sc} + N_{cp} - 1$. The forgetting factor α indicates how much we trust the previous samples. When $\alpha < 1$, the smaller the α , the fewer weights we put on the old samples than the recent ones.

The output weight $\hat{\mathbf{W}}_{out}^{(n)}(m)$ is recursively updated by

$$\hat{\mathbf{W}}_{out}^{(n)}(m) = \hat{\mathbf{W}}_{out}^{(n)}(m-1) + \mathbf{e}_n(m) \mathbf{v}_n^T(m), \quad (18)$$

where $\mathbf{e}_n(m) = \bar{\mathbf{x}}_n^t(m) - \hat{\mathbf{W}}_{out}^{(n)}(m-1) \mathbf{z}_n(m)$ is the error on the current sample m when estimated with weight matrix at the $(m-1)$ -th step, and $\mathbf{v}_n(m)$ is the gain vector computed by the following equation [28]:

$$\mathbf{v}_n(m) = \frac{\Phi_n^{-1}(m-1) \mathbf{z}_n(m)}{\alpha + \mathbf{z}_n^T(m) \Phi_n^{-1}(m-1) \mathbf{z}_n(m)}. \quad (19)$$

The $\Phi_n^{-1}(m) = (\sum_{m'=0}^m \alpha^{m-m'} \mathbf{z}_n(m') \mathbf{z}_n^T(m'))^{-1}$ is the inverse of the weighted correlation of $\mathbf{z}_n(m)$ and is recursively updated with

$$\Phi_n^{-1}(m) = \alpha^{-1} (\Phi_n^{-1}(m-1) - \mathbf{v}_n(m) [\mathbf{z}_n^T(m) \Phi_n^{-1}(m-1)]). \quad (20)$$

The output weight $\hat{\mathbf{W}}_{out}^{(n)}$ is determined by the weight learned at $(N_{sc} + N_{cp} - 1)$ -th step.

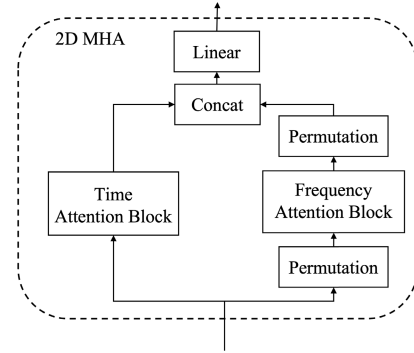


Fig. 5. 2D MHA module.

B. 2D MHA and StructNet With DF

In the frequency domain, the output of the RC $\hat{\mathbf{X}}_n^t$ is transformed to the frequency domain $\hat{\mathbf{X}}_n^f \in \mathbb{C}^{N_t \times N_{sc}}$, where the corresponding target output is $\mathbf{X}_n^f$. Denote $\hat{X}_n^f(n_t, n_{sc})$ and $X_n^f(n_t, n_{sc})$ as the (n_t, n_{sc}) -th entry of the $\hat{\mathbf{X}}_n^f$ and $\mathbf{X}_n^f$. The complex values are mapped to real values by $\mathbf{i}_n^f(n_t, n_{sc}) = f_R(\hat{X}_n^f(n_t, n_{sc}))$ and $\mathbf{o}_n^f(n_t, n_{sc}) = f_R(X_n^f(n_t, n_{sc}))$.

1) *Learning From Pilot Symbols*: The input first passes through the 2D MHA module. Unlike the MHA module in the Transformer that only captures feature correlations along the same dimension, the 2D MHA works in a two-dimensional manner, incorporating both time and frequency dimensions. As shown in Fig. 5, two attention blocks, including a time attention block and a frequency attention block, are utilized in the module to capture the time correlation and frequency correlation, respectively. The input to the time attention block is the real-valued frequency domain signal $\mathbf{I}^f(n_t) \in \mathbb{R}^{N_p \times 2N_{sc}}$, which is a concatenation of $\mathbf{i}_n^f(n_t, n_{sc})$ along the time and frequency dimensions. The input to the frequency attention block is the frequency domain signal $\tilde{\mathbf{I}}^f(n_t) \in \mathbb{R}^{N_{sc} \times 2N_p}$ obtained by a permutation of $\mathbf{I}^f(n_t)$. To facilitate the feature aggregation of each block, the value embedding dimension for each attention block is set to be equal to the input size. The output of the frequency attention block is permuted and then aggregated with the time attention block output through concatenation and linear projection. The final output of 2D MHA is added to the input through the residual connection and then passed through StructNet. Note that the 2D MHA is only employed when learning from pilot symbols to help obtain a better estimate of the data symbols. During the DF procedure, the network is updated on a symbol basis, which does not allow the module to capture the two-dimensional feature along the time dimension. Therefore, the 2D MHA is skipped during the DF procedure.

2) *Attention Loss*: The attention loss is designed for the frequency domain network to re-weight the training loss of different samples according to their confidence level. The idea behind it is to force the network to pay more attention to confident samples. The attention loss for each sample can be written as $\mathbf{a}_n^f(n_t, n_{sc}) \odot \ell(f_s(\mathbf{i}_n^f(n_t, n_{sc})), \mathbf{o}_n^f(n_t, n_{sc}))$, where $\mathbf{a}_n^f(n_t, n_{sc})$ is the corresponding weights of this training sample, $\odot$ is the Hadamard product, $\ell(\cdot)$ stands for the cross-

entropy loss, and $f_s(\cdot)$ represents the function approximated by the frequency domain network. When training with the pilot symbols, the confidence level for all the training samples is the same, and thus the weights for all the training samples are set to be 1, i.e., $\mathbf{a}_n^f(n_t, n_{sc}) = [1, 1]^T$. Note that the output of the frequency domain network is the detected data symbol along with its predicted probability. The predicted probability indicates how confident the network is in the detected data symbol. During the DF procedure, the training labels are the detected symbols. The attention weights are set as the predicted probability of detected symbols. It is true that the detected data symbol with a high predicted probability is not ensured to be correct. However, if the network predicts a high probability for the detected data symbol, it is more likely to be correct than the detected data symbol with a low predicted probability. When the detected data symbol has a high predicted probability, we put more weights on it than the detected symbol with a low predicted probability. In addition, only symbols with a predicted probability larger than η are used during the DF process. Therefore, the attention weights can be written as $\mathbf{a}_n^f(n_t, n_{sc}) = q(P(\tilde{\sigma}_n^f(n_t, n_{sc})))$, where $P(\tilde{\sigma}_n^f(n_t, n_{sc}))$ is the probability of the predicted symbol $\tilde{\sigma}_n^f(n_t, n_{sc})$ provided by the frequency domain network. The $q(\cdot)$ denotes the following non-linear function:

$$q(x) = \begin{cases} x, & \text{if } x \geq \eta \\ 0, & \text{otherwise} \end{cases}, \quad (21)$$

where η is the probability threshold. With the attention loss, the error propagation issue of the DF process can be alleviated by assigning weights to the loss based on the predicted probability and selecting samples with high confidence.

3) *DF With Data Symbols*: After training the 2D MHA and the StructNet with the pilot symbols, we adopt the DF mechanism to learn from the detected data symbols on an OFDM symbol basis. As the two-dimensional features do not exist when updating in a symbol-by-symbol fashion, the 2D MHA module is not used and only the StructNet is fine-tuned with the data symbols. Specifically, we obtain the detected data signals $\hat{\mathbf{X}}_n^t$ in the time domain utilizing the updated RC and transform it to frequency domain $\hat{\mathbf{X}}_n^f$. Then we test the StructNet updated by the $n - 1$ th data symbol with $\hat{\mathbf{X}}_n^f$ to get the estimated symbol $\tilde{\mathbf{X}}_n^f$ for the n th data symbol. The pair $(\hat{\mathbf{X}}_n^f, \tilde{\mathbf{X}}_n^f)$ is exploited as the training data to fine-tune the StructNet. Note that for $n = N_p$, i.e., the first data symbol, the training label is obtained by testing with the pilot-trained 2D MHA and the StructNet. For the rest data symbols, the training labels for fine-tuning are obtained only by the fine-tuned StructNet.

C. Summary of Symbol Detection Procedure

The symbol detection procedure of RC-AttStructNet-DF includes two parts: offline initialization and online training. We explain the offline initialization and summarize the online training in this subsection. The procedure is also shown in **Algorithm 1**.

Algorithm 1 Symbol Detection Procedure of RC-AttStructNet-DF

```

1: # Offline initialization
2: Prepare artificial training data
3: Train a binary classifier with artificial training data
4: Initialize binary classifier in StructNet with learned weights
5: # Online training
6: for Each OFDM subframe do
7:   # Learning from pilot symbols
8:   Prepare pilot training data  $D_{rc}$  with time domain received
      and transmitted pilot signals as defined in eq. (16)
9:   Train output weights of RC with  $D_{rc}$  using one-shot LS
      solution in eq. (3)
10:  Generate RC output (time domain RC pilot output) when
      tested on received pilot signals
11:  Transform time domain RC pilot output to frequency domain
      (frequency domain RC pilot output)
12:  Utilize frequency domain RC pilot output and transmitted
      pilot symbols to estimate effective channel using LMMSE
13:  Initialize PE layer of StructNet with estimated effective
      channel
14:  Prepare pilot training data for 2D MHA and StructNet with
      frequency domain RC pilot output and transmitted pilot symbols
15:  Train 2D MHA and StructNet with pilot training data using
      attention loss
16:  # DF with data symbols
17:  for OFDM symbol  $n = N_p : N - 1$  (data symbols) do
18:    Generate RC output ( $n$ -th time domain RC data output)
      by testing on  $n$ -th time domain received data signal
19:    Prepare training dataset for RC with  $n$ -th time domain
      received data signal and  $n$ -th time domain RC data output
20:    Update RC weights by RLS algorithm following eq. (17)
21:    Transform  $n$ -th time domain RC data output to frequency
      domain ( $n$ -th frequency domain RC data output)
22:    if  $n = N_p$  then
23:      Generate  $n$ -th detected data symbols with 2D MHA
      and StructNet by testing on  $n$ -th frequency domain
      RC data output
24:    else
25:      Generate  $n$ -th detected data symbols with StructNet
      by testing on  $n$ -th frequency domain RC data output
26:    end if
27:    Prepare training dataset for StructNet with  $n$ -th frequency
      domain RC data output and  $n$ -th detected data symbols
28:    Fine-tune StructNet weights with attention loss
29:  end for
30: end for

```

1) *Offline Initialization*: The binary classifier in StructNet is initialized with offline weights that are trained by artificially generated data. Note that this offline training does not require any prior knowledge of the channel, which differentiates our method from other work with offline initialization and online adaptation. Specifically, the training labels are randomly generated symbols $\mathbf{E} \in \mathcal{A}^{N_t \times N_{sp}}$, where $\mathcal{A}$ represents the set $\{-1, +1\}$ and N_{sp} is the number of offline training samples. The training inputs are the noise-contaminated symbols or the received signal $\mathbf{E} + \mathbf{G}$, where $\mathbf{G} \in \mathbb{R}^{N_t \times N_{sp}}$ is the Gaussian noise. In this way, the initial weights of the binary classifier are learned to conduct nearest neighbor mapping and facilitate the online training of the full StructNet, i.e., the PE layer and the binary classifier. Note that these offline weights, once trained, are fixed. The same weights are used for initialization when conducting online detection on different subframes.

TABLE III
TRAINING COMPLEXITY

Method	RC	Frequency domain network
RC-Struct	$\mathcal{O}(V(N_n + N_{\text{train}} + N_t)(N_n + N_r)N_{\text{train}})$	$\mathcal{O}(8N_tN_hN_{\text{ep}}N_{\text{sc}}N_p)$
RC-AttStructNet-DF	$\mathcal{O}(V(N_n + N_{\text{train}} + N_t)(N_n + N_r)N_{\text{train}} + ((N_n + N_r)N_t + 4(N_n + N_r)^2)N_{\text{test}})$	$\mathcal{O}(4N_pN_{\text{sc}}N_t(2N_k + N_{\text{sc}}) + 8N_t(N_{\text{ep}}N_p + N_{\text{epdf}}N_d)N_hN_{\text{sc}})$

TABLE IV
TESTING COMPLEXITY

Method	Complexity
LMMSE with LMMSE-CSI	$\mathcal{O}(N_pN_{\text{sc}}^2N_a^2 + N_dN_{\text{sc}}(N_a^3 + N_a^2 + N_a))$
LMMSE with LMMSE-Interpolation	$\mathcal{O}((N_pN_{\text{sc}}^2 + 7N_dN_{\text{sc}})N_a^2 + N_dN_{\text{sc}}(N_a^3 + N_a^2 + N_a))$
SD with LMMSE-Interpolation	$\mathcal{O}((N_pN_{\text{sc}}^2 + 7N_dN_{\text{sc}})N_a^2 + N_dN_{\text{sc}}M^{N_a}(2N_a^2 + 2N_a - 1))$
RC-Struct	$\mathcal{O}(V(N_n + N_r)(N_nN_{\text{test}} + N_t) + 4N_tN_hN_{\text{sc}}N_d)$
RC-AttStructNet-DF	$\mathcal{O}(V(N_n + N_r)(N_nN_{\text{test}} + N_t) + 4N_{\text{sc}}N_t(2N_k + N_{\text{sc}} + N_hN_d))$

2) *Online Training*: The time domain RC network and the frequency domain network are learned separately. When learning from pilot symbols, RC is first trained with the time domain received and transmitted pilot signals. Then after the training of RC, the output of RC is transformed into the frequency domain. The frequency domain network is learned by taking frequency domain RC output as the input and the transmitted pilot symbols in the frequency domain as the training label. When learning from detected data symbols with the DF mechanism, the network weights are updated in a symbol-by-symbol manner. For each data symbol, we generate the RC output of the data symbol by testing on the time domain received data signal. This RC output is employed as the training label to re-train RC with the RLS algorithm. The time domain RC output is then converted to frequency domain. The detected data symbol is obtained by testing the frequency domain network on frequency domain RC output. Similarly, this detected data symbol is utilized as the training label for fine-tuning frequency domain network.

It is noteworthy that the combination of RC in the time domain and the frequency domain network is critical in our design. This is because even though StructNet is designed to learn from the training samples efficiently, training StructNet along with 2D MHA still requires a relatively large amount of training data. Since RC can efficiently decouple different data streams and equalize the channel in time domain, as shown in our previous work [19], [20], the classification task in frequency domain becomes much more accessible to tackle after the processing of RC.

VI. COMPLEXITY ANALYSIS

This section analyzes the computational complexity of RC-AttStructNet-DF. The complexity is compared with RC-Struct [21], LMMSE detector, and sphere decoding (SD) approach. In the analysis, we mainly consider the computation cost of matrix multiplication and pseudo-inverse, as the costs for matrix addition and element-wise operation are negligible compared to these main factors. For ease of discussion, we denote the number of training and testing samples in the time domain as $N_{\text{train}} = (N_{\text{cp}} + N_{\text{sc}})N_p$ and $N_{\text{test}} = (N_{\text{cp}} + N_{\text{sc}})N_d$. As the complexities of RC-Struct have been provided in [21], we summarize the conclusions in Tab. III

and Tab. IV and mainly focus on the complexity analysis of RC-AttStructNet-DF. V denotes the number of cascaded RCs.

In the time domain, RC is first learned with the pilot symbols and then updated with the detected data symbols. As the DF is utilized, the training complexity will be larger than RC-Struct due to the adoption of the RLS update procedure with extra training on the data symbols. The complexity for training RC with pilot symbols is the same as RC-Struct, which is $\mathcal{O}(V(N_n + N_{\text{train}} + N_t)(N_n + N_r)N_{\text{train}})$. The RLS procedure on the data symbols has three steps to update the output weights. The complexity for updating $\hat{\mathbf{W}}_{\text{out}}^{(n)}(m)$ in eq. (18) for each sample is $\mathcal{O}((N_n + N_r)N_t)$. The update of $\mathbf{v}_n(m)$ in eq. (19) has a complexity of $\mathcal{O}((N_n + N_r)^2 + N_n + N_r) \approx \mathcal{O}((N_n + N_r)^2)$. The complexity for updating $\Phi_n^{-1}(m)$ in eq. (20) is $\mathcal{O}(3(N_n + N_r)^2)$. Then the complexity for all the samples with the RLS approach is $\mathcal{O}(((N_n + N_r)N_t + 4(N_n + N_r)^2)N_{\text{test}})$. Thus, the training complexity in time domain is $\mathcal{O}(V(N_n + N_{\text{train}} + N_t)(N_n + N_r)N_{\text{train}} + ((N_n + N_r)N_t + 4(N_n + N_r)^2)N_{\text{test}})$. At the testing stage, time domain RC in RC-AttStructNet-DF has the same complexity as RC-Struct, which is $\mathcal{O}(V(N_n + N_r)(N_nN_{\text{test}} + N_t))$.

The frequency domain network is composed of the 2D MHA module and the StructNet. For simplicity, we assume the time and frequency attention block in the 2D MHA adopts the same key embedding dimension N_k . Then the complexity for the time attention block is $\mathcal{O}((4N_kN_pN_{\text{sc}} + 4N_pN_{\text{sc}}^2)N_t)$. The complexity for frequency attention block is $\mathcal{O}((4N_kN_pN_{\text{sc}} + 4N_pN_{\text{sc}})N_t)$. The output linear project has a complexity of $\mathcal{O}(8N_pN_{\text{sc}}N_t)$. Thus, the total complexity for training 2D MHA with pilot symbols is $\mathcal{O}(4N_pN_{\text{sc}}N_t(2N_k + N_{\text{sc}} + 3)) \approx \mathcal{O}(4N_pN_{\text{sc}}N_t(2N_k + N_{\text{sc}}))$. During the DF procedure, 2D MHA is only adopted for testing the first data symbol, resulting in a complexity of $\mathcal{O}(4N_{\text{sc}}N_t(2N_k + N_{\text{sc}}))$.

For StructNet, it consists of a PE layer and a binary classifier. The PE layer conducts an element-wise multiplication and thus the complexity is ignored here. Denote N_h as the number of neurons in the input layer and N_{ep} as the number of training epochs. When training with pilot symbols, the training complexity of StructNet is the same as the frequency domain network of RC-Struct, which is $\mathcal{O}(8N_tN_hN_{\text{ep}}N_{\text{sc}}N_p)$. When DF is adopted, the extra training complexity is $\mathcal{O}(8N_tN_hN_{\text{epdf}}N_{\text{sc}}N_d)$, where N_{epdf} is the

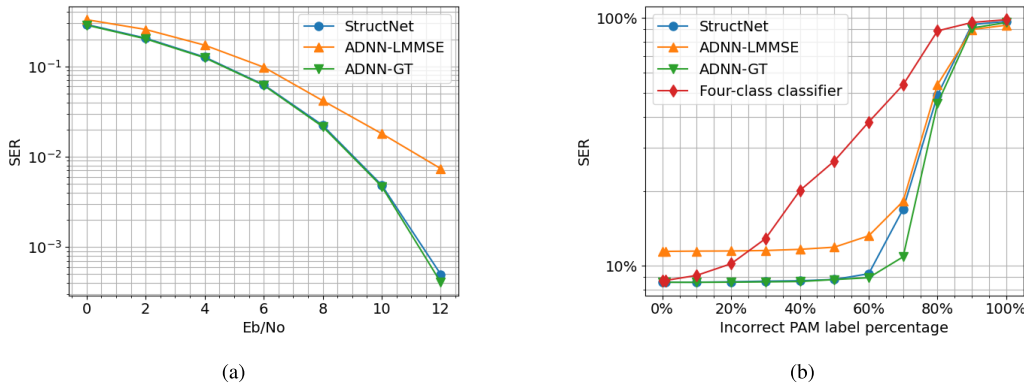


Fig. 6. Comparison of SER. (a) with different E_b/N_o 's in dB. (b) with different percentages of incorrect PAM labels.

number of fine-tuning epochs for DF. Thus, the total complexity is $\mathcal{O}(8N_t(N_{\text{ep}}N_p + N_{\text{epdf}}N_d)N_hN_{sc})$. As the number of testing samples in the frequency domain is $N_tN_{sc}N_d$, the testing complexity is $\mathcal{O}(4N_tN_hN_{sc}N_d)$.

The LMMSE approach is a low-complexity linear detector that is widely used in communication systems. The SD detector [29] is a non-convex solver that approaches the optimal maximum likelihood (ML) detection, which has high detection complexity and thus is rarely used in practice. As channel estimation is needed as the input to both methods, we adopt LMMSE for the underlying channel estimation. As our previous work [19] has analyzed the complexities of both methods with the LMMSE channel estimation in details, we summarize the conclusions in Tab. IV. The “LMMSE-CST” indicates that the channel estimates are obtained by only utilizing the pilot symbols. Meanwhile, the “LMMSE-Interpolation” means that the channel estimates are interpolated over the data symbols using the pilot-estimated CSI. To simplify the expression, we assume that the number of antennas satisfies $N_a = N_r = N_t$.

The analysis shows that RC-AttStructNet-DF has higher training and testing complexity than RC-Struct due to the additional 2D MHA module and DF procedure. However, the training and testing complexities of these two approaches are still in the same order of magnitude. Compared with the conventional approaches, RC-AttStructNet-DF has higher complexity than the LMMSE approach due to the extra training stage, and a lower complexity than the SD detector.

VII. TOY EXPERIMENT: MIMO SYSTEM WITH GAUSSIAN CHANNEL

In this section, we provide a toy experiment in a MIMO system with the Gaussian channel to analyze the properties of StructNet. We start with analyzing the effectiveness of the PE layer and then empirically show the robustness of StructNet to incorrect labels.

A. Experimental Setting

In the toy experiment, we assume the classifier has sufficient data and time to be trained, and the PE layer is initialized with an inaccurate LMMSE estimated channel. To satisfy such an

assumption, the number of training samples is set to be 1000, among which 4 samples are utilized for LMMSE channel estimation, and the rest 996 samples are used for training the classifier. The trained network is tested with 3000 samples. For simplicity, we test in a 2×2 MIMO with 4-PAM modulation. 100 Gaussian channel realizations are tested. The channels are selected to have condition numbers smaller than 1.5 to mimic the setting with dynamic transmission modes, where data is transmitted in channels with small condition numbers. In Sec. VIII, we remove such assumptions and evaluate our method in more realistic 3GPP 3D channels.

B. Effectiveness of PE Layer

We compare three approaches: 1) ADNN-GT: The ADNN with perfect channel knowledge; 2) ADNN-LMMSE: The ADNN with LMMSE estimated channel; 3) StructNet: our introduced approach. The classifiers in all these three networks are comprised of two linear layers connected with the hyperbolic tangent non-linear function, and trained with the cross-entropy loss. In Fig. 6 (a), we plot the symbol error rate (SER) as a function of bit energy to noise ratio (E_b/N_o). Compared with ADNN-LMMSE, StructNet can achieve better performance. The performance gain is more significant with a relatively high E_b/N_o . In addition, StructNet is shown to have comparable SER performance with ADNN-GT, where the perfect channel knowledge is used. The results indicate that even if the PE layer starts from an inaccurate initialization, StructNet can achieve comparable performance with the approach exploiting the ground truth CSI.

C. Experiments of Training With Incorrect Labels

In Sec. IV-B, we analyzed the reason why StructNet is robust to incorrect PAM labels. In this section, we conduct the experiment to show how the performance of StructNet is affected by different percentages of incorrect training labels. In the experiment, we randomly select a certain percentage of PAM samples to be incorrect. The performance is evaluated under 5 dB E_b/N_o . Besides the two methods mentioned above, we also compare the performance with a four-class classifier. For a fair comparison, the four-class classifier also adopts two linear layers and the hyperbolic tangent non-linear function,

except that the output layer is of size 4. The results of SER versus the incorrect PAM label percentage are shown in Fig. 6 (b). As opposed to the general four-class classifier that is significantly affected by incorrect labels, StructNet performs reasonably well with even 70% incorrect PAM labels. This is because the 70% incorrect PAM labels only contribute to 35% incorrect binary labels, making it easier to handle by the network. The results are consistent with our analysis and demonstrate the ability of StructNet to combat the incorrect training samples. Such a property allows StructNet to mitigate the error propagation issue inherent in DF-based approaches, making it suitable to be applied with the DF mechanism.

VIII. EVALUATION WITH 3GPP-3D CHANNEL

In this section, we show the performance of RC-AttStructNet-DF both in the MIMO-OFDM system and the massive MIMO-OFDM system. Unlike the setting in RC-Struct [21], the experiments are conducted at a user speed of 30 km/h. We compare the introduced approach with the conventional model-based methods and state-of-the-art learning-based strategies.

A. Experimental Setting

In the experiments, the number of subcarriers is set to be $N_{sc} = 512$ and the CP length is $N_{cp} = 32$. Each subframe has a total of $N = 20$ OFDM symbols, among which $N_p = 4$ OFDM symbols are the training pilot and $N_d = 16$ OFDM symbols are the data symbols. Note that only 4 pilot symbols are used as the training data for each subframe, which is different from other learning-based approaches that exploit a large amount of training data. The wireless channels are generated following the 3GPP 3D MIMO model [17] with the QuaDRiGa simulator [30]. The user speed is set as 30 km/h, which is different from the setting in RC-Struct with a speed of 5 km/h. In addition, gray coding is adopted in the experiments.

In terms of the setting of RC, the number of neurons is $N_n = 16$, and the number of layers is $V = 1$. In [19], it is shown that the utilization of a sliding window for the input to RC can increase the short-term memory capacity of RC. Following this work, a sliding window of size 32 is utilized to the input. Note that all RC-based approaches compared in this work adopt the same sliding window for a fair comparison. In the 2D MHA, the key embedding dimension for the time and frequency attention block is set as 216 and 8, respectively. For the StructNet in the frequency domain, the input layer has 128 neurons, and the output layer has 2 neurons. In StructNet, the offline weights of the binary classifier are trained with 2000 randomly generated symbols for 1000 epochs. As choosing E_b/N_o for training the offline weights does not provide any significant performance gain and is also not practical, the E_b/N_o for each training sample is randomly chosen from 0 dB to 15dB to obtain relatively generic offline weights. During online training, we adopt an alternative training strategy, which updates the PE layer and the binary classifier separately. Specifically, we first update the binary classifier and fix the PE layer. Then the PE layer is updated with the weights of the binary classifier fixed. The

total number of training epochs is set to be 20. In addition, to reduce the computation complexity, nine resource block groups (RBGs) are combined to train a single network. The probability threshold for attention loss is set as $\eta = 0.5$. It means that we select the detected symbols that the network predicts to have over 50% chance to be correct. In other words, if the network thinks the detected data symbol has less than or equal to a 50% of the chance being correct, the detected symbol is less likely to be correct and thus we do not use it in the DF procedure.

B. BER Comparison in the MIMO-OFDM System and the Massive MIMO-OFDM System

We compare the following approaches: (1) *LMMSE+LMMSE-CSI*: The LMMSE-based symbol detector with LMMSE estimated CSI; (2) *LMMSE+LMMSE-Interpolation*: The LMMSE-based symbol detector using interpolated LMMSE CSI; (3) *SD+LMMSE-Interpolation*: The non-convex symbol detector that performs ML detection with SD approach utilizing interpolated LMMSE CSI [29]; (4) *MMNet*: The MMNet network proposed in [12], where the network for each subcarrier has been trained for 500 iterations; (5) *RC-Struct*: The RC-based approach using LMMSE estimated shifting parameter in the frequency domain [21]; (6) *RC-AttStructNet-DF*: The introduced method in this paper. Note that the “LMMSE-CSI” refers to that the LMMSE channel estimates only use the pilot symbols. The “LMMSE-Interpolation” means that the channel estimates are interpolated over data symbols using the pilot estimated CSI.

We conduct experiments in two system settings: the MIMO-OFDM system and the massive MIMO-OFDM system. In the MIMO-OFDM system, the number of transmit antennas is set as $N_t = 4$ and the number of receive antennas is $N_r = 4$. In Fig. 7, we show the BER plot for QPSK, 16 QAM, 64 QAM, respectively.¹ The results show that all the learning-based approaches and the SD method outperform LMMSE when the CSI is obtained by using only the pilot symbols for all the tested modulation orders. With the interpolated CSI, the BER of the LMMSE detection scheme decreases, as the channel estimates become more accurate when interpolated over the data symbols. As exhibited in Fig. 7 (b) and Fig. 7 (c), the SD approach has better performance than the RC-Struct and MMNet in the high E_b/N_o regime when 16 QAM and 64 QAM modulation orders are used. However, in the low E_b/N_o regime, the performance of SD becomes worse than RC-Struct and MMNet due to inaccurate channel estimates. The reason is that the channel estimates in the low E_b/N_o regime are less precise than in the high E_b/N_o regime, leading to performance degradation. These observations indicate that the performance of the conventional approaches LMMSE and SD heavily relies on the accuracy of the CSI estimation. The inherent error of the CSI estimation

¹Note that the BER of RC-AttStructNet-DF (without channel coding) ranges from 3% to 14%, which lies within the typical BER range specified by the 3GPP 5G NR [31], [32]. For instance, the user equipment (UE) channel quality indicator (CQI) calculation is based on a target block error rate (BLER) of 10% (after channel coding) [31], and radio link monitoring out-of-sync BLER is set to be 10% [32].

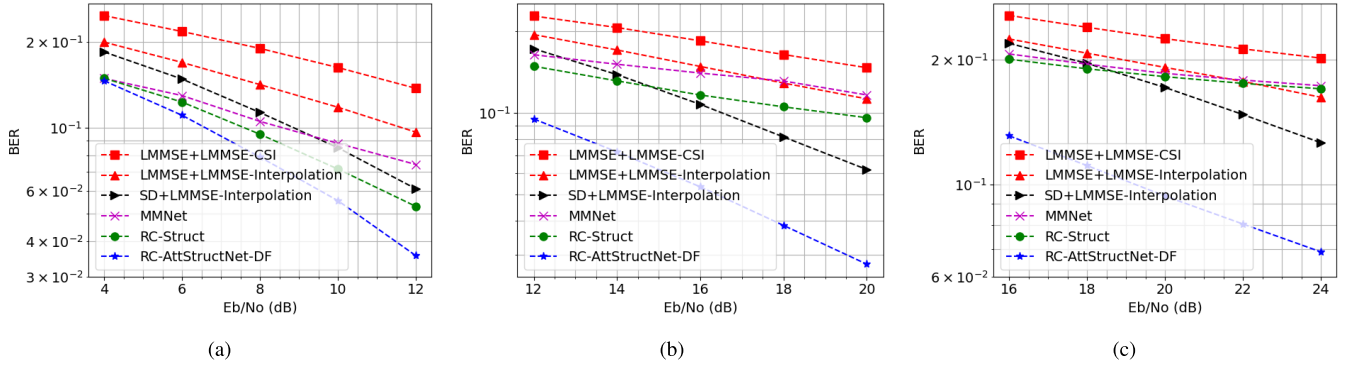


Fig. 7. BER comparison in the MIMO-OFDM system. (a) QPSK (b) 16 QAM (c) 64 QAM.

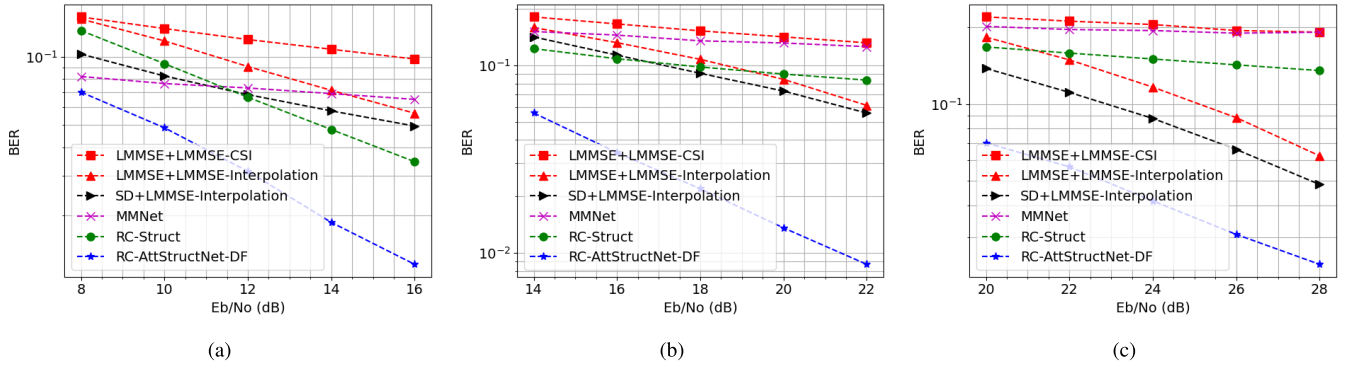


Fig. 8. BER comparison in the massive MIMO-OFDM system. (a) QPSK (b) 16 QAM (c) 64 QAM.

can jeopardize the detection performance. On the other hand, as a learning-based approach that does not rely on explicit channel estimation, RC-AttStructNet-DF is not affected by such impairments and is shown to have outstanding performance gain over conventional methods for all the tested modulation orders.

Regarding the learning-based approaches, in Fig. 7, we can see that the RC-AttStructNet-DF consistently outperforms the MMNet algorithm and the RC-Struct method under different scenarios. The reason is that for MMNet, it requires a larger amount of training data than the setup in this paper to learn the network weights. As an online over-the-air scenario is adopted in our evaluation, the MMNet learned by the limited amount of training data suffers from the model overfitting problem, resulting in performance degradation. Different from MMNet, by embedding the structural knowledge of the MIMO-OFDM system, RC-AttStructNet-DF can be learned with limited training data in an online fashion. Furthermore, both the RC-Struct and the MMNet only learn from the pilot symbols. Due to the relatively high user mobility, the neural network weights only trained by the pilot symbols are not sufficient to track the changes of the channel over data symbols, and thus have an unsatisfactory detection performance when testing on the data symbols. Instead, RC-AttStructNet-DF dynamically updates the network weights according to the channel variation with the data symbols. With the specially designed architecture and the dynamic adaptation, better performance is achieved by the RC-AttStructNet-DF.

In the massive MIMO-OFDM system, we test in an uplink scenario with 4 transmit antennas and 64 receive antennas. In particular, at the transmitter side, the number of scheduled UE is 2, where each UE has 2 transmit antennas. At the receiver side, the base station (BS) is equipped with a rectangular planar array that has 8 azimuth antennas and 8 elevation antennas. Fig. 8 shows the BER performance in the massive MIMO-OFDM system with QPSK, 16 QAM, and 64 QAM. The same trend holds as in the MIMO-OFDM system, where RC-AttStructNet-DF achieves the lowest BER. The results further demonstrate the advantages of RC-AttStructNet-DF over the other methods under different scenarios.

C. BER Comparison With Nonlinear Distortion

In this section, we perform the experiment when PA distortion is applied to the input signal. The PA model $g(x) = \frac{x}{\left[1 + \left(\frac{|x|}{x_{\text{sat}}}\right)^{2\rho}\right]^{0.5\rho}}$ is adopted to introduce channel distortion [33], where x is the input transmitted signal, x_{sat} is the PA saturation level, and ρ measures the smoothing parameter. We define the input back-off (IBO) as the ratio between PA's saturation power to the input power. The input signal is distorted when the peak-to-average-power ratio (PAPR) of the input signal is higher than IBO. We adopt $x_{\text{sat}} = 1$ and $\rho = 3$ and set the non-linear region as the case when IBO is smaller than 6.5 dB. The experiments are conducted in the 4×4 MIMO-OFDM system with 16 QAM modulation.

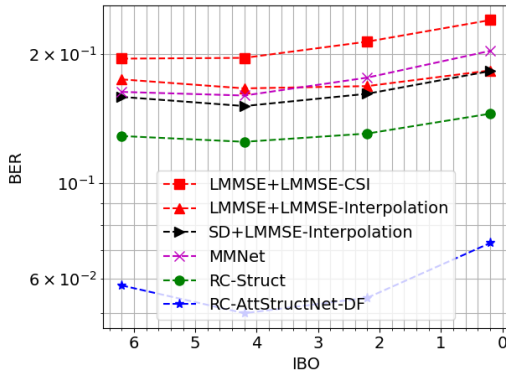


Fig. 9. BER comparison in the nonlinear region.

Fig. 9 shows the performance when the non-linear distortion exists. The performance of the conventional schemes, such as the SD and LMMSE, is highly affected by the system's nonlinearity. Specifically, the BER of the SD method increases quickly when the IBO reduces, as the estimated CSI becomes less accurate with stronger signal distortion. As the distortion level increases, the performance of the MMNet approach also degrades due to the linear assumption for its system model. Furthermore, RC-based approaches perform better than the conventional approaches when IBO is low, indicating that the RC-based approaches are better at combating the nonlinearity. More importantly, RC-AttStructNet-DF achieves the best performance among all the schemes, demonstrating its generalization ability in different cases.

D. Effectiveness of Attention Mechanism, PE Layer, and DF

In this section, we demonstrate the effectiveness of incorporating the attention mechanism, PE layer, and DF. The experiments are conducted in the MIMO-OFDM system with 4 transmit antennas and 4 receive antennas. The modulation order is set as 64 QAM. In Fig. 11, we compare the BER performance of four methods: (1) RC-Struct; (2) RC-Struct-DF; RC-Struct with DF; (3) RC-StructNet-DF: the approach with PE layer and DF; (5) RC-AttStructNet-DF: our introduced approach with the attention mechanism, PE layer, and DF. As shown in Fig. 11, RC-Struct-DF outperforms RC-Struct, which indicates the effectiveness of using DF when structural information is incorporated in the network. By comparing the performance of RC-StructNet-DF with RC-Struct-DF, we can see that the BER performance is further boosted when the PE layer is adopted in the frequency domain. The results suggest that the introduced PE layer in StructNet can further facilitate the DF mechanism and improve the detection performance by allowing the network to dynamically update the network parameters according to channel variations. Moreover, RC-AttStructNet-DF achieves an additional performance gain over RC-StructNet-DF, demonstrating that the attention mechanism is a valuable addition to our network.

E. BER Comparison With Practical Pilot Pattern

While our previous discussions are all based on the pilot structure where the first N_p OFDM symbols are all pilots,

in this section, we show that our introduced approach can also be applied to a scattered pilot pattern specified by the 3GPP standard. We consider the MIMO pilot pattern shown in Fig. 10 following the 3GPP 5G NR [34], [35]. The pilot resource elements (REs) are colored in yellow and the data REs are colored in blue. The white REs with cross markers represent the empty pilot symbols. As illustrated in Fig. 10 (a), for conventional approaches, empty pilot symbols are transmitted. In addition, pilots are set to be orthogonal among different antenna ports. The empty and orthogonal settings of the pilot are utilized to eliminate pilot interference and ensure a more accurate MIMO channel estimation. For our introduced learning-based method, we adopt the pilot structure in Fig. 10 (b), where all the pilots are randomly generated. It is noteworthy that the pilot pattern in Fig. 10 (b) has the same training overhead as the pilot structure in Fig. 10 (a). The difference is that we try to avoid pilot interference using the empty and orthogonal pilots for conventional approaches, while we keep the pilot interference with the pilot pattern for learning-based methods. The reason is that, for conventional detectors that rely on channel estimation, the received pilots should be free of interference to obtain a more accurate estimated CSI. However, for learning-based methods, the neural network needs to learn the situation when interference exists to avoid the mismatch between the training stage and the testing stage. More detailed discussions about the pilot pattern design have been provided in our previous work [19].

The experiment is conducted in the 4×4 MIMO-OFDM system with 16 QAM modulation. The total number of OFDM symbols is set as 14 following the 3GPP 5G NR standard [34], [35]. The training overhead of this scattered pilot pattern is approximately 19%. Note that the block pilot pattern, where there are 20 OFDM symbols in total and the first 4 OFDM symbols are pilots, has a training overhead of 20%, which also satisfies the pilot occupancy requirement specified in [34] and [35]. With the scattered pilot pattern, the RC-AttStructNet-DF network is first trained with the pilot REs to obtain the initial weights. Then the DF is conducted for the full subframe, including the OFDM symbol with pilot REs, in a symbol-by-symbol manner. During the DF procedure, when re-training with the detected OFDM symbol that has pilot REs, the pilot positions are filled up with the ground truth pilots instead of detected pilots to generate training labels. As displayed in Fig. 12, our introduced RC-AttStructNet-DF approach outperforms both the conventional LMMSE and SD methods with this practical scattered pilot pattern. More importantly, the results further demonstrate the generalization ability of RC-AttStructNet-DF and its potential to be adopted in a realistic setting.

F. BER Comparison With Conventional Methods Using Decision-Directed Channel Estimation

In this section, we conduct the performance comparison with conventional approaches when the RLS-based decision-directed (DD) channel estimation [36] is adopted. Specifically, the first four pilot OFDM symbols are utilized to obtain the initial channel estimation using the RLS scheme. Then

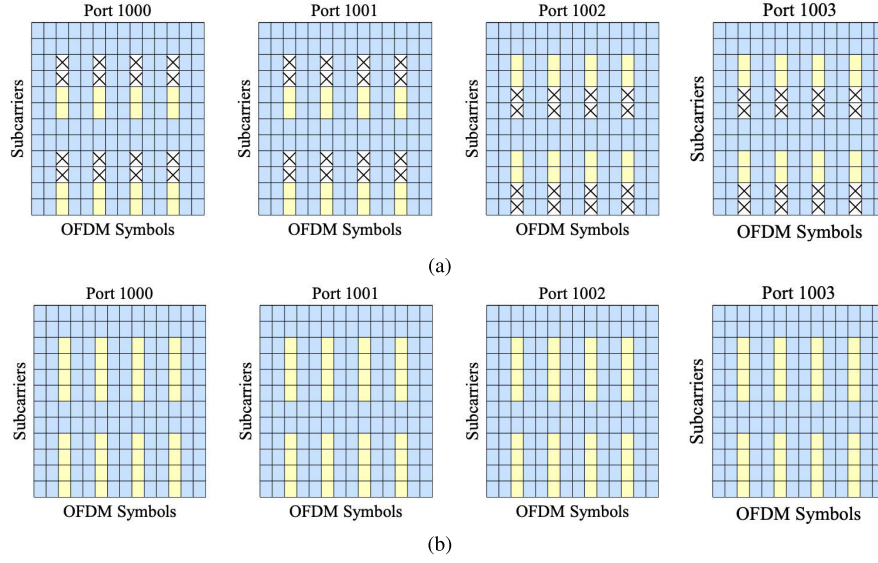


Fig. 10. The MIMO scattered pilot pattern in one resource block (RB). (a) Conventional approaches. (b) RC-AttStructNet-DF.

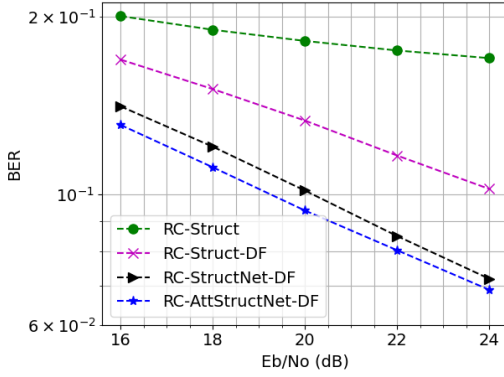


Fig. 11. BER for testing effectiveness of different modules.

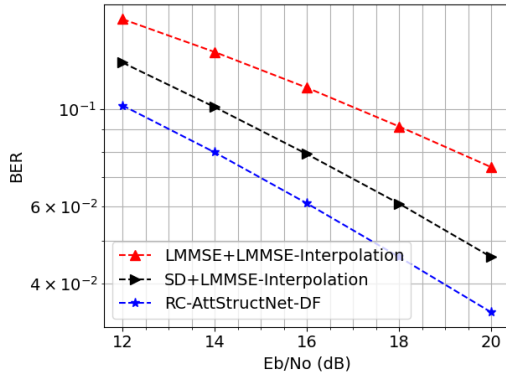


Fig. 12. BER comparison with the scattered pilot pattern.

the subsequent CSI corresponding to the data symbols is obtained by treating the detected symbols as training data and estimated iteratively using the RLS approach. The BER performance is provided in Fig. 13. The LMMSE detector and SD detector with this DD channel estimation are referred to as “LMMSE+DD-CSI” and “SD+DD-CSI”, respectively. The experiment is performed under the 4×4 MIMO-OFDM system with 16 QAM modulation. As shown in Fig. 13, by dynamically estimating the channel of data

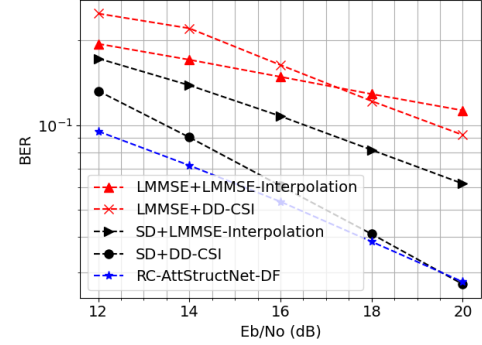


Fig. 13. BER comparison with conventional schemes using DD-CSI.

symbols, “LMMSE+DD-CSI” has better performance than the LMMSE detector with interpolated LMMSE CSI in the high E_b/N_o regime. However, due to the error propagation caused by the inaccurate data symbol detection, the performance of “LMMSE+DD-CSI” is degraded in mid to low E_b/N_o regimes. The performance of the SD detector is boosted by the DD channel estimates when compared with the SD detector utilizing the interpolated LMMSE CSI. Compared with conventional approaches, our introduced RC-AttStructNet-DF achieves better performance than “LMMSE+DD-CSI” across different E_b/N_o ’s and outperforms “SD+DD-CSI” in the mid to low E_b/N_o regimes. We further compare the performance when the PA distortion is applied to the input signal. As indicated in Fig. 14, the performance of the “LMMSE+DD-CSI” and “SD+DD-CSI” deteriorate when signal distortion becomes stronger, especially for the “SD+DD-CSI”. On the other hand, RC-AttStructNet-DF is less affected by the nonlinearity within the system and exhibits better performance than the conventional approaches.

G. Empirical Complexity of Symbol Detection Approaches

To see the performance-complexity trade-off more straightforwardly, we compare the empirical complexity of different

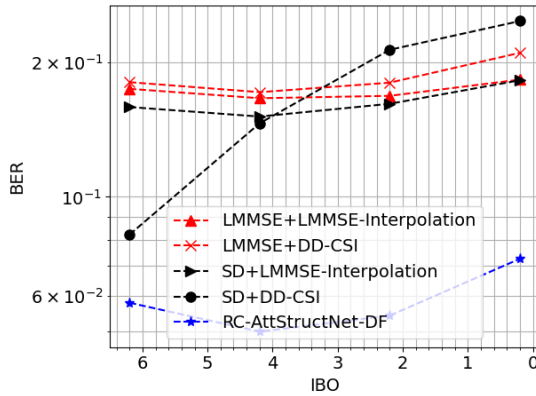


Fig. 14. BER comparison with adopting DD-CSI in nonlinear region.

TABLE V
CPU RUN TIME OF SYMBOL DETECTION METHODS

Method	Training time (Sec.)	Total processing time (Sec.)
LMMSE+LMMSE-CSI	-	0.12
LMMSE+LMMSE-Interpolation	-	7.57
SD+LMMSE-Interpolation	-	152.06
RC-Struct	17.55	18.27
RC-AttStructNet-DF	26.67	29.57
MMNet	3014.52	3025.66

methods under the same setting for Fig. 7 (c). Specifically, we show CPU run time for different methods in the setting of the MIMO-OFDM system with 4 transmit antennas and 4 receive antennas and 64 QAM modulation. In Tab. V, we present both the training time and the total processing time for a subframe of different methods, where the total processing time includes both the training and testing time for a subframe. Tab. V indicates that RC-AttStructNet-DF has a longer total processing time than LMMSE-based approaches. However, RC-AttStructNet-DF can achieve a 65% and 57% BER reduction over the LMMSE with LMMSE-CSI and LMMSE with LMMSE-Interpolation under 24 dB E_b/N_o , respectively, as shown in Fig. 7 (c). While RC-AttStructNet-DF takes a longer total processing time than RC-Struct, the CPU run time of these two approaches are still on the same scale. With the same scale of the total processing time, RC-AttStructNet-DF performs significantly better than RC-Struct. On the other hand, when compared with the SD method, RC-AttStructNet-DF processes in a shorter amount of time and is also demonstrated to have a better performance. More importantly, when compared with the state-of-the-art learning-based approach MMNet, RC-AttStructNet-DF processes over 100 times faster, which demonstrates the potential of RC-AttStructNet-DF to be adopted for online detection.

IX. CONCLUSION

In this paper, we introduce an online attention-based approach, RC-AttStructNet-DF, for conducting MIMO-OFDM symbol detection on a subframe basis. The approach employs reservoir computing (RC) in the time domain. The frequency domain network consists of the 2D MHA module along with a structure-based network StructNet, which is learned with an attention loss. The 2D MHA is exploited to capture the

time and frequency correlations of the signal. The StructNet is designed to mitigate the error propagation of the DF approach and facilitate the DF mechanism. With the StructNet, the adopted DF mechanism further enhances detection performance by learning from detected data symbols and dynamically tracking channel changes within a subframe. Extensive experiments in 3GPP 3D channels demonstrate the effectiveness of RC-AttStructNet-DF in detection under different scenarios with the BER performance and the effectiveness of different modules.

REFERENCES

- [1] H. Ye, G. Y. Li, and B. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, Feb. 2018.
- [2] Z. Zhao, M. C. Vuran, F. Guo, and S. D. Scott, "Deep-waveform: A learned OFDM receiver based on deep complex-valued convolutional networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2407–2420, Aug. 2021.
- [3] Q. Chen, S. Zhang, S. Xu, and S. Cao, "Efficient MIMO detection with imperfect channel knowledge—A deep learning approach," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2019, pp. 1–6.
- [4] M. Honkala, D. Korpi, and J. M. J. Huttunen, "DeepRx: Fully convolutional deep learning receiver," *IEEE Trans. Wireless Commun.*, vol. 20, no. 6, pp. 3925–3940, Jun. 2021.
- [5] X. Lyu, W. Feng, and N. Ge, "Deep neural network-based symbol detection for highly dynamic channels," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2020, pp. 1–6.
- [6] Y. Liao, N. Farsad, N. Shlezinger, Y. C. Eldar, and A. J. Goldsmith, "Deep neural network symbol detection for millimeter wave communications," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.
- [7] X. Yi and C. Zhong, "Deep learning for joint channel estimation and signal detection in OFDM systems," *IEEE Commun. Lett.*, vol. 24, no. 12, pp. 2780–2784, Dec. 2020.
- [8] R. Shafin, L. Liu, V. Chandrasekhar, H. Chen, J. Reed, and J. C. Zhang, "Artificial intelligence-enabled cellular networks: A critical path to beyond-5G and 6G," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 212–217, Apr. 2020.
- [9] L. Liu, R. Chen, S. Geirhofer, K. Sayana, Z. Shi, and Y. Zhou, "Downlink MIMO in LTE-advanced: SU-MIMO vs. MU-MIMO," *IEEE Commun. Mag.*, vol. 50, no. 2, pp. 140–147, Feb. 2012.
- [10] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2554–2564, May 2019.
- [11] H. He, C. Wen, S. Jin, and G. Y. Li, "A model-driven deep learning network for MIMO detection," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Nov. 2018, pp. 584–588.
- [12] M. Khani, M. Alizadeh, J. Hoydis, and P. Fleming, "Adaptive neural signal detection for massive MIMO," *IEEE Trans. Wireless Commun.*, vol. 19, no. 8, pp. 5635–5648, Aug. 2020.
- [13] H. He, C. Wen, S. Jin, and G. Y. Li, "Model-driven deep learning for MIMO detection," *IEEE Trans. Signal Process.*, vol. 68, pp. 1702–1715, 2020.
- [14] S. Shi, Y. Cai, Q. Hu, B. Champagne, and L. Hanzo, "Deep-unfolding neural-network aided hybrid beamforming based on symbol-error probability minimization," *IEEE Trans. Veh. Technol.*, vol. 72, no. 1, pp. 529–545, Jan. 2023.
- [15] K. Kang, Q. Hu, Y. Cai, G. Yu, J. Hoydis, and Y. C. Eldar, "Mixed-timescale deep-unfolding for joint channel estimation and hybrid beamforming," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 9, pp. 2510–2528, Sep. 2022.
- [16] H. He, C. Wen, S. Jin, and G. Y. Li, "Deep learning-based channel estimation for beamspace mmWave massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 852–855, Oct. 2018.
- [17] *Study on 3D Channel Model for LTE*, document TR 36.873, 3GPP, 2015.
- [18] S. S. Mosleh, L. Liu, C. Sahin, Y. R. Zheng, and Y. Yi, "Brain-inspired wireless communications: Where reservoir computing meets MIMO-OFDM," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4694–4708, Oct. 2018.
- [19] Z. Zhou, L. Liu, and H. Chang, "Learning for detection: MIMO-OFDM symbol detection through downlink pilots," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 3712–3726, Jun. 2020.

- [20] Z. Zhou, L. Liu, S. Jere, J. Zhang, and Y. Yi, "RCNet: Incorporating structural information into deep RNN for online MIMO-OFDM symbol detection with limited training," *IEEE Trans. Wireless Commun.*, vol. 20, no. 6, pp. 3524–3537, Jun. 2021.
- [21] J. Xu, Z. Zhou, L. Li, L. Zheng, and L. Liu, "RC-struct: A structure-based neural network approach for MIMO-OFDM detection," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 7181–7193, Sep. 2022.
- [22] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.
- [23] Z. Zhou, L. Liu, V. Chandrasekhar, J. Zhang, and Y. Yi, "Deep reservoir computing meets 5G MIMO-OFDM systems in symbol detection," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 1, pp. 1266–1273.
- [24] L. Li, L. Liu, J. Zhang, J. D. Ashdown, and Y. Yi, "Reservoir computing meets Wi-Fi in software radios: Neural network-based symbol detection using training sequences and pilots," in *Proc. 29th Wireless Opt. Commun. Conf. (WOCC)*, May 2020, pp. 1–6.
- [25] Z. Zhou, S. Jere, L. Zheng, and L. Liu, "Learning for integer-constrained optimization through neural networks with limited training," in *Proc. NeurIPS Wkshps Learn. Meets Combinat. Algorithms*, Dec. 2020, pp. 1–12.
- [26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
- [27] B. Farhang-Boroujeny, *Adaptive Filters: Theory and Applications*. Hoboken, NJ, USA: Wiley, 2013.
- [28] H. Jaeger, "Adaptive nonlinear system identification with echo state networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 15, 2002, pp. 1–8.
- [29] A. Ghasemmehdi and E. Agrell, "Faster recursions in sphere decoding," *IEEE Trans. Inf. Theory*, vol. 57, no. 6, pp. 3530–3536, Jun. 2011.
- [30] S. Jaeckel, L. Raschkowski, K. Börner, and L. Thiele, "QuaDRiGa: A 3-D multi-cell channel model with time evolution for enabling virtual field trials," *IEEE Trans. Antennas Propag.*, vol. 62, no. 6, pp. 3242–3256, Jun. 2014.
- [31] *5G; NR; Physical Layer Procedures for Data*, document TS 38.214, Rev. 16.6.0, 3GPP, 2021.
- [32] *5G; NR; Requirements for Support Radio Resource Manage.*, document TS 38.133, Rev. 17.2.0, 3GPP, 2021.
- [33] C. Rapp, "Effects of HPA-nonlinearity on a 4-DPSK/OFDM-signal for a digital sound broadcasting signal," in *Proc. ESASP*, vol. 332, 1991, pp. 179–184.
- [34] *5G; NR; Physical Channels and Modulation*, document TS 36.211, Rev. 16.2.0, 3GPP, 2020.
- [35] *5G; NR; Multiplexing Channel Coding*, document TS 36.212, Rev. 16.2.0, 3GPP, 2020.
- [36] E. Karami and M. Shiva, "Decision-directed recursive least squares MIMO channels tracking," *EURASIP J. Wireless Commun. Netw.*, vol. 2006, no. 1, pp. 1–10, Dec. 2006.



Jiarui Xu (Graduate Student Member, IEEE) received the B.S. degree in telecommunication engineering from the Beijing Institute of Technology, Beijing, China, in 2017, and the M.S. degree in electrical and computer engineering from the University of Michigan, Ann Arbor, MI, USA, in 2019. She is currently pursuing the Ph.D. degree with the Bradley Department of Electrical and Computer Engineering (ECE), Virginia Tech, Blacksburg, VA, USA. Her research interests include deep learning, machine learning, and their applications in wireless communications.



Lianjun Li received the B.S. degree in telecommunications engineering from Zhejiang University, Hangzhou, China, the M.S. degree in electrical engineering from The University of Texas at Dallas, Richardson, TX, USA, and the Ph.D. degree in electrical engineering from Virginia Tech, Blacksburg, VA, USA. Currently, he is a Staff Research Engineer with Samsung Research America. His research interests include applying reinforcement learning and deep learning techniques to wireless communications.



Lihong Zheng (Fellow, IEEE) received the B.S. and M.S. degrees from Tsinghua University, Beijing, China, in 1994 and 1997, respectively, and the Ph.D. degree from The University of California, Berkeley, CA, USA, in 2002. Since 2002, he has been with the Department of Electrical Engineering and Computer Sciences, Massachusetts Institute of Technology, Cambridge, MA, USA, where he is currently a Professor of electrical engineering and computer sciences. His research interests include information theory, wireless communications, and statistical inference. He was a recipient of the Eli Jury Award from UC Berkeley in 2002, the IEEE Information Theory Society Paper Award in 2003, the NSF CAREER Award in 2004, and the AFOSR Young Investigator Award in 2007.



Lingjia Liu (Senior Member, IEEE) received the B.S. degree in electronic engineering from Shanghai Jiao Tong University and the Ph.D. degree in electrical and computer engineering from Texas A&M University. Currently, he is a Professor and the Bradley Senior Faculty Fellow with the ECE Department, Virginia Tech (VT), and also the Director of the Wireless@Virginia Tech, a center focusing on wireless technology. He spent more than four years with the Mitsubishi Electric Research Laboratory (MERL) and the Standards and Mobility Innovation Laboratory, Samsung Research America (SRA), where he received the Global Samsung Best Paper Award in 2008 and 2010. He was leading Samsung's efforts on multiuser MIMO, CoMP, and HetNets in 3GPP LTE/LTE-advanced standards. His general research interests include enabling technologies for 5G-advanced/6G networks, including incorporating domain knowledge into machine learning for wireless networks, massive MIMO, massive MTC communications, and mm-wave communications. His research received eight best paper awards. In 2021, he received the VT College of Engineering Dean's Award for Excellence in Research.