DeepMux: Deep-Learning-Based Channel Sounding and Resource Allocation for IEEE 802.11ax

Pedram Kheirkhah Sangdeh and Huacheng Zeng
Department of Computer Science and Engineering, Michigan State University, East Lansing, MI USA

***Abstract*—MU-MIMO and OFDMA are two key techniques in IEEE 802.11ax standard. Although these two techniques have been intensively studied in cellular networks, their joint optimization in Wi-Fi networks has been rarely explored as OFDMA was introduced to Wi-Fi networks for the first time in 802.11ax. The marriage of these two techniques in Wi-Fi networks creates both opportunities and challenges in the practical design of MAC-layer protocols and algorithms to optimize airtime overhead, spectral efficiency, and computational complexity. In this paper, we present DeepMux, a deep-learning-based MU-MIMO-OFDMA transmission scheme for 802.11ax networks. DeepMux mainly comprises two components: deep-learning-based channel sounding (DLCS) and deep-learning-based resource allocation (DLRA), both of which reside in access points (APs) and impose no computational/communication burden on Wi-Fi clients. DLCS reduces the airtime overhead of 802.11 protocols by leveraging the deep neural networks (DNNs). It uses uplink channels to train the DNNs for downlink channels, making the training process easy to implement. DLRA employs a DNN to solve the mixed-integer resource allocation problem, enabling an AP to obtain a near-optimal solution in polynomial time. We have built a wireless testbed to examine the performance of DeepMux in real-world environments. Our experimental results show that DeepMux reduces the sounding overhead by** 62.0%∼90.5% **and increases the network throughput by** 26.3%∼43.6%.

***Index Terms*—IEEE 802.11ax, machine learning, deep neural network, Wi-Fi, multi-user MIMO, OFDMA, channel sounding, resource allocation**

I. Introduction

After two decades of evolution from its genesis, Wi-Fi technology has become the dominant carrier of the Internet traffic [1] and penetrated every aspect of our lives. With the continuous proliferation of the Internet-based applications, Wi-Fi market is growing at an unprecedented rate, and more than four billion Wi-Fi devices have shipped in 2019 alone [1]. To serve the large number of Wi-Fi devices and meet their high data rate demands, Wi-Fi networks are evolving from 802.11n/ac to 802.11ax so that a Wi-Fi access point (AP) is capable of utilizing the spectrum more efficiently and accommodating more Wi-Fi clients at the same time. Compared to the carrier-sense-based 802.11n/ac, 802.11ax features centralized resource allocation and fine-grained inter-device synchronization. With these two features, it introduces orthogonal frequency-division multiple access (OFDMA) and uplink multi-user multiple-input multiple-output (MU-MIMO) techniques for the first time.

Although OFDMA and MU-MIMO has been well studied in cellular networks (see Table I), their joint optimization in Wi-Fi networks remains scarce because OFDMA is introduced to Wi-Fi networks in 802.11ax for the first time. Given that cellular and Wi-Fi networks have different PHY (physical) and MAC (medium access control) layers, and that base stations (BSs) and APs have very different computational power, the MU-MIMO-OFDMA transmission schemes designed for cellular networks may not be suited for Wi-Fi networks, necessitating research efforts to innovate the MU-MIMO-OFDMA design for 802.11ax networks. Particularly, the MU-MIMO-OFDMA transmission in 802.11ax faces two challenges. *First*, to perform downlink MU-MIMO transmissions, an AP needs to have channel state information (CSI) for the construction of beamforming filters so that it can concurrently send independent data streams to multiple Wi-Fi clients on the same Resource Unit (RU). However, existing 802.11 channel sounding protocols are notorious for their large airtime overhead, which significantly compromises the throughput gain of MU-MIMO. Therefore, a low-overhead channel sounding protocol is needed. *Second*, the marriage of MU-MIMO and OFDMA largely expands the optimization space of resource allocation at an 802.11ax AP, making it infeasible to pursue an optimal resource allocation solution in real time due to the limited computational power of APs. Therefore, a low-complexity, yet efficient, algorithm is needed for an AP to solve the resource allocation problem.

In this paper, we study the channel sounding and resource allocation problems for downlink transmissions in an 802.11ax Wi-Fi network, where an AP serves many stations (STAs) on a set of pre-defined RUs jointly using MU-MIMO and OFDMA techniques. We assume that the AP is equipped with multiple antennas, while each STA is equipped with one antenna. In such an 802.11ax network, we propose a practical scheme, called DeepMux, to enhance the efficiency of downlink MU-MIMO-OFDMA transmissions by leveraging recent advances in deep learning (DL). DeepMux addresses the above two challenges using deep neural networks (DNNs), and it mainly comprises the following two key components: i) DL-based channel sounding (DLCS), and ii) DL-based resource allocation (DLRA). Both of them reside in APs and impose no computational/communication burden to the STAs.

To reduce the channel sounding overhead, DLCS in DeepMux compresses the frequency-domain CSI during the feedback procedure by leveraging the compression capability of DNNs. Specifically, instead of reporting CSI on all the grouped tones, each STA only reports the quantized CSI on a *small*

The authors are with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824 (Corresponding author: H. Zeng). This work was supported in part by the NSF grants CNS-2100112 and CNS-2113618.

number of tones to the AP. Based on the limited CSI, the AP infers CSI over all tones using well-trained DNNs. Particularly, the AP takes advantage of channel reciprocity and uses uplink CSI, which is easy to obtain, to train the DNNs for downlink CSI, making the training process easy to conduct.

To obtain a near-optimal resource allocation solution in real time at the AP, DLRA in DeepMux employs a DNN to solve a mixed-integer non-linear programming (MINLP) optimization problem. Specifically, DLRA decouples the complex resource allocation optimization problem into two sub-problems: RU assignment and power allocation. A DNN is then employed to compute a sub-optimal solution to the RU assignment sub-problem. Once RU assignment is determined, the original MINLP problem degrades to a linear programming (LP) problem, which is easy to solve.

The contributions of this paper are summarized as follows.

- We have designed DLCS, a DL-based channel sounding protocol for 802.11ax networks. DLCS employs an online training process and requires no efforts from STAs. Numerical results show that DLCS is capable of reducing the channel sounding overhead by $62.0\%{\sim}90.5\%$ without sacrificing CSI feedback accuracy.
- We have designed DLRA, a DL-based resource allocation algorithm for 802.11ax APs to perform efficient downlink transmissions. Numerical studies show that DLRA is capable of yielding a sub-optimal solution to MINLP resource allocation problems in polynomial time.
- By combining DLCS and DLRA, we have designed DeepMux to enable efficient downlink MU-MIMO-OFDMA transmissions in 802.11ax networks. We have evaluated DeepMux on a wireless testbed. Experimental results show that DeepMux improves network throughput by $26.3\%{\sim}43.6\%$ compared the greedy utilization of DoF by strongest STAs on each RU.

The remainder of this paper is organized as follows. Section II surveys the related work. Section III describes the existing MU-MIMO protocols and discusses the underlying challenges. Section IV explains DeepMux in nutshell. Section V delineates the proposed DLCS protocol, and Section VI describes the DLRA algorithm. Section VII presents our experimental results. Section VIII concludes the paper.

II. Related Work

We focus our literature review on channel sounding and resource allocation in both Wi-Fi and cellular networks.

A. Channel Sounding

Channel Sounding for Wi-Fi: The sounding overhead issue in Wi-Fi networks has been in focal point of view since the accommodation of MU-MIMO in IEEE 802.11 standards. Existing research efforts have been invested to tackle this issue by optimizing channel sounding parameters [2]–[4], seeking new channel sounding paradigms [5], [6], or compressing CSI frames [7], [8]. As the pioneering trials of reducing sounding overhead, research efforts in [2]–[4] have exploited the semi-static nature of Wi-Fi networks to adaptively reduce the frequency of channel sounding and avoid unnecessary sounding overhead. Implicit channel sounding has also been studied for rectifying sounding overhead [5], [6]. Although implicit channel sounding can significantly lower the overhead, it requires extra hardware for channel calibration and thus may not be amenable to low-cost Wi-Fi networks. DeepMux is orthogonal to these works as DLCS neither manipulates the channel sounding frequency nor employs implicit channel sounding.

[7] and [8] are two prior efforts that reduce the channel sounding overhead by compressing CSI in the frequency domain. However, these two efforts require coordination from Wi-Fi clients to fully or partially compress CSI. In contrast, DLCS runs solely on Wi-Fi routers and requires no coordination from Wi-Fi clients. Simply put, DLCS is transparent to Wi-Fi clients. DLCS also differs from these two works in terms of computational complexity. Specifically, [7] and [8] require Wi-Fi clients to estimate CSI for all frequency tones while DLCS requires Wi-Fi clients to estimate CSI only for a small number of tones.

Learning-Based Channel Sounding in Cellular Networks: Sounding overhead is also a critical problem in cellular networks. Temporal correlation [9]–[11] and spatial correlation [9] have been harvested to remove the redundancy of CSI and reduce the airtime overhead of CSI acquisition. DeepMux differs from these works as it focuses on the frequency domain. Frequency-domain correlation of CSI has been studied in [12] and [13] to reduce the channel sounding overhead in cellular networks. DeepMux differs from these works because DLCS is transparent to users (i.e., imposing no computation on users). In addition, CSI in cellular networks is very different from that in Wi-Fi networks. DeepMux is meticulously tailored for Wi-Fi networks. Finally, most prior works are limited to theoretical investigations and numerical evaluations while DeepMux takes into account incumbent Wi-Fi protocols and has been validated in practical indoor wireless environments.

B. Resource Allocation

Table I summarizes existing resource allocation schemes in cellular and Wi-Fi networks. Clearly, DeepMux differs from existing works in terms of objective, network scenario, transmission mode, or computational complexity. In what follows, we elaborate the existing studies and point out the differences between DeepMux and these works.

Resource Allocation for Wi-Fi Networks: Recently, [14] has studied downlink OFDMA in wireless local area networks (WLANs) and showed that its performance is highly dependent on the resource assignment strategies at APs. This problem has been followed in [15], with the objective of improving the fairness among users. DLRA differs from the proposed resource allocation scheme in [15] as it focuses on pursuing a sub-optimal resource allocation scheme with a low computational complexity. [16] has considered the throughput maximization under the assumption that a user can be assigned to at most one RU and offered a solution for both uplink and downlink transmissions. Compared to [16], DLRA expands the problem scope by allowing multiple RUs to serve a user and also by allowing an RU to serve multiple users

TABLE I: A summary of resource allocation schemes in Wi-Fi and cellular networks, where n denotes the number of active users served by an AP or a BS.

	Objective				Network		Mode		MU-MIMO	Polynomial complexity
	sum-rate	Fairness	Latency	Energy	Wi-Fi	Cellular	Uplink	Downlink		
DeepMux	✓				✓			✓	✓	$\mathcal{O}\left(n^{2.5}\right)$
[15]	✓	✓			✓			✓		$\mathcal{O}\left(n^{3}\right)$
[16]	✓				✓		✓	✓		
[17], [18]	✓				✓			✓	✓	
[19]			✓		✓		✓			
[20]		✓			✓		✓			$\mathcal{O}\left(n^{3}\right)$
[21]	✓				✓		✓			
[22]	✓		✓			✓		✓	✓	
[23]				✓		✓		✓		$\mathcal{O}\left(n^{3}\right)$
[24]				✓		✓		✓	✓	
[25]				✓		✓		✓	✓	$\mathcal{O}\left(n^{2.5}\right)$
[26]–[28]	✓					✓		✓	✓	

concurrently. [17] and [18] are the only works considering downlink MU-MIMO-OFDMA in WLANs. However, these two works employ greedy iterative algorithms to compute a feasible solution. In contrast, DLRA employs learning-based approach and offers a solution in polynomial time. [19]–[21] studied resource allocation in uplink OFDMA WLANs, which is not the scope of our work.

Resource Allocation in Cellular Networks: Since there are many research results of resource allocation in cellular networks, we focus our review on MIMO-OFDMA techniques. [22] has studied the resource allocation problem under latency constraint. However, the complexity of the proposed solution is prohibitively large. [23] has studied the resource allocation problem with the objective of enhancing energy efficiency. The authors has proposed an algorithm with polynomial-time complexity. However, it only works for single-user MIMO-OFDMA networks. [24] and [25] have investigated the resource allocation problem for MU-MIMO-OFDMA cellular networks and proposed low-complexity algorithms to compute the solutions. However, these two works focus on maximizing energy efficiency. In contrast, DeepMux aims to maximize network throughput. [26]–[28] have explored downlink MU-MIMO-OFDMA transmissions in different network scenarios. These research efforts have proposed greedy algorithms to pursue optimal solutions for maximizing network throughput. DeepMux is very different from these works in terms of network settings and computational complexity.

III. Problem Description

Consider an 802.11ax network comprising a multi-antenna AP and many single-antenna STAs. Denote N_{ap} as the number of antennas on the AP. Denote N_{sta} as the number of STAs in the network. We consider a dense network where $N_{\mathrm{sta}} > N_{\mathrm{ap}}$. In 802.11ax standard, OFDMA and MU-MIMO techniques have been included for efficient communications between the AP and its serving STAs. Fig. 1 shows the four possible RU configurations when the network works on 20 MHz bandwidth. As the figure shows, the total number of valid tones is 242, and an RU could consists of 26, 52, 106, or 242 tones. When MU-MIMO is enabled, an RU can serve multiple STAs, depending on the channel condition, data traffic, and network setting. In the downlink transmissions, in order for an AP to serve multiple STAs per RU, it needs to first perform channel sounding to obtain the CSI and then construct the spatial filters for beamforming. By doing so, independent data streams can be delivered to different STAs simultaneously. In this process, CSI is crucial. In what follows, we first present the existing channel sounding protocol and then state our design objectives.

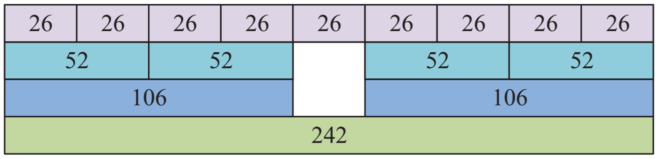


Fig. 1: Four different RU configurations over 20 MHz as specified in IEEE 802.11ax [29].

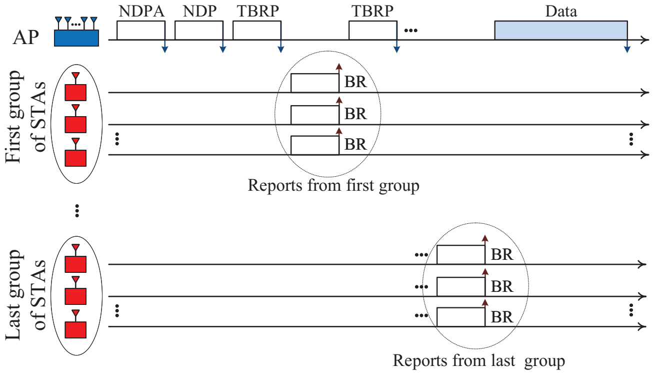


Fig. 2: Existing channel sounding protocol in IEEE 802.11ax.

A. 802.11 Channel Sounding Protocol in Nutshell

Fig. 2 shows the channel sounding protocol specified in 802.11ax, and we elaborate on it in the following.

Announcement: The AP initiates the channel sounding procedure by broadcasting a Null Data Packet Announcement (NDPA) frame, which contains the addresses of intended STAs. Then, the AP sends out a Null Data Packet (NDP) frame for STAs to estimate the downlink channels between themselves and the AP.

Channel Estimation: Each STA leverages the preamble in the NDP frame to estimate the complex-valued channel vectors between the AP and itself. Reporting the raw channel vectors to the AP, however, entails too much airtime overhead. To reduce the airtime overhead, each STA employs *Givens rotations* and *tone grouping* to pre-process its estimated channel vectors. The pre-processing leads to a CSI compression in both spatial and spectral domains.

Spatial compression: In its general form, the spatial compression includes a series of Givens rotations, pre-multiplications, and post-multiplications applied to the right singular vectors of a channel matrix to extract its spatial information [29]–[31]. Each rotation or pre-multiplication is realized by an angle, which stores a part of spatial information [32]. On each tone, two sets of angles will be generated: N_ψ ψ-type angles from Givens rotations and N_ϕ ϕ-type angles from pre-multiplications, where $N_\psi = N_\phi = \left(2N_{\mathrm{ap}}N_r - N_r^2 - N_r\right)/2$ and N_r is the number of the STA's antennas in general case (we assumed $N_r = 1$ in this paper). For notional simplicity, we denote these two sets over all tones as $\boldsymbol{\Psi} = \{\psi_{i,k}\}_{\forall i,k}$ and $\boldsymbol{\Phi} = \{\phi_{i,k}\}_{\forall i,k}$, where i is the angle index ($1 \leq i \leq N_\psi$), k

is the tone index ($1 \leq k \leq N_{tone}$), and N_{tone} is the number of tones.

Generally speaking, $\psi_{i,k} \in [0, \pi/2)$ and $\phi_{i,k} \in [0, 2\pi)$. The angles will be quantized before being sent to the AP. In 802.11 standards, two types of quantization are specified for feedback:

- *Feedback type 0* uses 5 bits for each angle in $\mathbf{\Psi}$ and 7 bits for each angle in $\mathbf{\Phi}$.
- *Feedback type 1* uses 7 bits for each angle in $\mathbf{\Psi}$ and 9 bits for each angle in $\mathbf{\Phi}$.

Tone Grouping: As Wi-Fi networks typically work in indoor scenarios for short-range communications, their coherence bandwidth tends to be large. Hence, *tone grouping* has been employed to bond N_g tones. In 802.11ax standard [29], $N_g = \{1, 4, 16\}$. Particularly, $N_g = 1$ means that no grouping is employed. Also, $N_g = 16$ is only allowed with feedback type 1.

Beamforming Report (BR): The BR frames carry the quantized angles ($\mathbf{\Psi}$ and $\mathbf{\Phi}$) from each STA to the AP. These frames are also used to carry the channel strength information (average SNR and SNR deviation for each group of tones) from each STA to the AP. Based on the reported SNR information, the AP manages available spectral and power resources to serve STAs.

Polling: Polling is a mechanism to coordinate the report process among STAs. Once all STAs have prepared their BR frames, the AP sends trigger beamforming report poll (TBRP) frames sequentially. Each TBRP frame coordinates a group of STAs to send their BR frames through uplink MU-MIMO as illustrated in Fig. 2. The AP decodes the BR frames and identifies the sender of each report using the MAC address in the corresponding frame. After polling all the groups, the AP obtains information required for downlink MU-MIMO transmission.

B. Design Objectives and Challenges

The objectives of this work are to design and evaluate a practical, yet efficient, downlink MU-MIMO-OFDMA transmission scheme for 802.11ax networks. Towards these objectives, we face the following two challenges.

Challenge 1 – Channel Sounding Overhead: Channel sounding is crucial for beamforming in downlink MU-MIMO transmissions. However, the existing channel sounding protocol in Fig. 2 entails a large airtime overhead and significantly compromises the throughput gain of MU-MIMO. For instance, consider an AP with 8 antennas and a single-antenna STA working on 160 MHz bandwidth. Even with the tone grouping, the angles information in a single report could be as large as 7.0 kB[1], which is far beyond a maximum transmission unit (2.3 kB) in WLANs [33]. This means that a BR frame in Fig. 2 can take more than 3 packets for CSI feedback. Such a large airtime overhead not only consumes network bandwidth but it also ruins the freshness of CSI for beamforming.

Challenge 2 – Joint Resource Allocation: The marriage of MU-MIMO and OFDMA creates a joint resource allocation problem for the AP, which involves RU assignment for users and power allocation for MIMO streams. This problem is complicated as it crosses spectral and power domains. Solving the resource allocation problem is time-constrained as the coherence of wireless channels degrades over time. It is therefore important for an AP to have a low-complexity algorithm that can find an efficient resource allocation solution in real time. A classical approach for solving this problem is to first formulate the problem as an optimization problem and then employ existing optimization solvers to compute the optimal solution. This approach, however, is infeasible in practice due to the high computational complexity from an exhaustive search over RU assignment instances. For example, consider a small 802.11ax network where a 4-antenna AP serves 6 single-antenna STAs over four 52-tone RUs on 20 MHz bandwidth. We formulate the resource allocation problem as an MINLP optimization problem and employ CVX package to solve it for a given RU assignment. Our observation is that it takes up to 342 minutes to find an optimal solution with search over $2^{23.3}$ RU assignment instances. Such a large delay makes resource allocation infeasible for practical use and urges us to devise a low-complexity resource allocation mechanism.

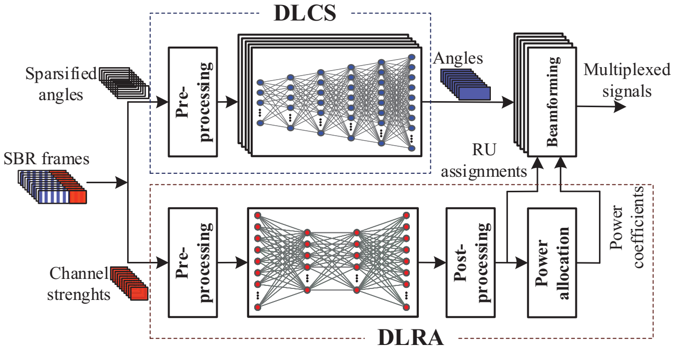


Fig. 3: The overview of DeepMux.

IV. Overview of DeepMux

In this section, we present an overview of DeepMux, which leverages recent advances in DNNs to address the challenges for downlink MU-MIMO-OFDMA transmissions in 802.11ax networks. Fig. 3 shows a high-level structure of DeepMux. It mainly comprises two components: DLCS and DLRA. In what follows, we present the basic idea of these two components.

A. Basic Idea of DLCS

DLCS is an enhanced 802.11 channel sounding protocol aiming to reduce the sounding overhead. Its design is based on the following two observations: i) wireless channels in local area networks are highly correlated in the frequency domain; and ii) tone grouping in the current 802.11 sounding protocol is not an efficient approach for feedback compression. Motivated by the success of DNNs for image compression, we propose to use DNNs to reduce the channel sounding overhead in the CSI feedback process. Specifically, instead of reporting CSI over a large number of tones, each STA only reports CSI over a small number of tones. Based on the reported CSI over

[1] In this case, $N_\psi = N_\phi = 7$ and feedback type 1 is used over 498 groups of tones. Representation of angles requires $55,776$ bits ≈ 7.0 kB.

sparse tones, the AP attempts to infer the CSI over all tones using DNNs.

While the idea is straightforward, an important question is how to train the DNNs so that they can infer the full CSI based on the limited feedback. For this question, one solution is that the AP asks every STA to report a large amount of CSI over all tones at the beginning and uses the large amount of CSI to train the DNNs. This solution, however, imposes heavy computational and communication burdens on STAs, and thus is not amenable to implementation. To circumvent this issue, we use uplink CSI, instead of downlink CSI, for the training of DNNs. This is because uplink and downlink channels have the same profile in the frequency domain, thanks to the channel reciprocity [34]. In other words, uplink and downlink channels bear the same shape over frequency domain even without channel calibration, making it possible for DNNs to learn the downlink frequency-domain CSI correlation using uplink CSI samples in the absence of channel calibration.

Additionally, an AP can easily obtain uplink CSI over all tones. Obtaining uplink CSI requires no effort from STAs, making the training process transparent to the STAs. Whenever an AP receives packets from STAs, it can measure the uplink channel based on the packets' preamble. We note that, different from prior channel reciprocity applications, channel calibration is not needed for our application. Details of DLCS are presented in Section V.

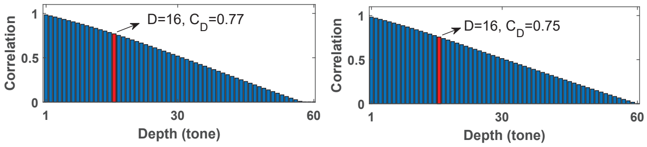
B. Basic Idea of DLRA

The marriage of MU-MIMO and OFDMA creates a challenge for an 802.11ax-enabled AP to optimally allocate the available spectral and power resources in a reasonable amount of time. To address this challenge, DeepMux formulates the resource allocation problem as an optimization problem. In its original form, the optimization problem is an MINLP problem, where its binary variables correspond to RU assignment sub-problem and its continuous variables correspond to power allocation sub-problem. DeepMux approaches the MINLP problem by reformulating it into a mixed-integer linear programming (MILP) problem. Unlike an MINLP problem, an MILP problem can be systematically solved in two steps: i) an organized search mechanism over discrete instances of the feasible region (RU assignment instances), and ii) an interior-point algorithm that solves the convex sub-problem (power allocation) for a given RU assignment.

Given that MILP is NP-hard in general, we take advantage of recent advances in DNNs to determine the optimal RU-assignment in the first step. Specifically, DeepMux employs a DNN to compute the values for the binary optimization variables in the MILP problem. Such a DNN is trained offline, in a supervised manner, using the SNR reports from STAs, as shown in Fig. 3. After the binary variables (corresponding to the RU assignment sub-problem) are determined, the MILP problem degrades to a linear programming problem, which is easy to solve. Details of DLRA are presented in Section VI.

V. DLCS: A low-overhead channel sounding

DLCS enhances the 802.11 channel sounding protocol in Fig. 2 by reducing the airtime consumed by BR frames. This is done through *sparsification* of $\mathbf{\Psi}$ and $\mathbf{\Phi}$ angles in the frequency domain. That is, each STA reports CSI angles over a few tones, and the AP infers the CSI angles for all tones based on the sparsified feedback using DNNs.



(a) Correlation of angles in $\mathbf{\Psi}$. (b) Correlation of angles in $\mathbf{\Phi}$.

Fig. 4: Spectral correlation of angles in $\mathbf{\Psi}$ and $\mathbf{\Phi}$.

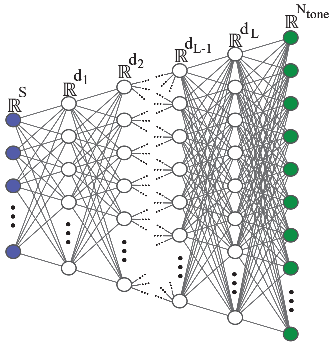


Fig. 5: DNNs' structure at the AP for inferring $\mathbf{\Psi}$ and $\mathbf{\Phi}$ based on limited feedback.

Before diving into DLCS, we first take a look at the frequency-domain correlation of CSI angles. We collected $50,000$ $\mathbf{\Psi}$ and $\mathbf{\Phi}$ samples in an office environment to measure the frequency-domain correlation. For a sequence $\mathbf{x} \in \mathbb{R}^{1\times L}$, we define C_D as its correlation at depth D by letting:

$$C_D = \mathbb{E}_m \left[\frac{\mathbf{x}_{(m+1:m+D)}\mathbf{x}^T_{(m+D+1:m+2D)}}{\|\mathbf{x}_{(m+1:m+D)}\|\|\mathbf{x}_{(m+D+1:m+2D)}\|} \right], \quad (1)$$

where $\mathbf{x}_{(i:j)} \triangleq [x_i, x_{i+1}, \cdots, x_j]$ with x_i being the ith element in $\mathbf{x}$, and $(\cdot)^T$ is transpose operator. Fig. 4 shows the correlation of the collected CSI angles at different tone depths. It can be seen that, when the tone depth is greater than 16 (i.e., $D > 16$), the correlation is still considerable for both $\mathbf{\Psi}$ and $\mathbf{\Phi}$ angles. This means that, grouping the angles over N_g tones (simply by averaging operation) cannot fully harvest such a significant correlation for compression purpose. On the other hand, tone grouping may lead to an inaccurate feedback when $N_g > 16$. DLCS is a more sophisticated compression approach to reduce the sounding overhead by exploiting inter-tone CSI correlation.

In what follows, we first present the settings of DNNs and then elaborate on their training (exploration) and sparsification (exploitation) phases separately.

A. DNNs Settings

As shown in Fig. 5, DLCS employs DNNs at the AP to infer full CSI angles based on a sparsified feedback. One DNN is used for the angles in $\mathbf{\Psi}$ and another DNN is used for the angles in $\mathbf{\Phi}$. The dimension of input layer is S, corresponding to the quantized CSI angles over S tones. The value of S is selected through experimental studies, which will be shown shortly. The DNNs have N_{tone} neurons on the output layer, corresponding to the inferred CSI angles over all tones (e.g., $N_{\text{tone}} = 234$ for all the nine 26-tone RUs over 20 MHz bandwidth). DNNs have multiple hidden layers, say L hidden layers. The dimension of the ith hidden layer is d_i. Each hidden layer is fully-connected, followed by a batch-normalization layer to speed up the training convergence [35]. Rectified linear unit (ReLU) activation function is used for each layer. Since the DNNs are designed for interpolation purpose, they are in an enlarging trapezoid shape.

B. Training Phase

As we explained before, the AP does not require STAs to report a large amount of downlink CSI angles for training DNNs because doing so imposes heavy computational and communication burdens on STAs. Instead, the AP uses its estimated uplink channels to calculate CSI angles and train the DNNs by taking advantage of wireless channel reciprocity. Since the DNNs focus only on learning the frequency-domain properties of CSI, channel calibration is not necessary to compensate the response difference between Tx and Rx RF chains.

Using the uplink CSI to train the DNNs have two benefits. *First*, it is easy for an AP to collect a large amount of samples for training purpose. As long as an STA sends a packet, the AP can estimate the uplink channel and use it for generating angles and training DNNs. Simply put, the AP requires zero effort to obtain dataset for training DNNs. *Second*, it tends to offer better training results as uplink CSI does not suffer from tone grouping and quantization errors. If the AP wants to use downlink CSI for training DNNs, quantization of the estimated downlink CSI at STAs is needed to facilitate the feedback. This introduces quantization error and degrades the training performance. In contrast, using uplink CSI for training purpose does not suffer from this issue.

In what follows, we describe the operations of DNNs training at the AP. No extra effort is needed at the STAs.

Data Collection: AP and STAs work in their ordinary mode. Whenever the AP receives a packet, it decodes the packet and records its estimated uplink channel on all tones. Then, the AP performs spatial compression on the estimated uplink channel over every tone, as specified in 802.11 standards [29] to collect CSI angles (i.e., $\mathbf{\Psi}$ and $\mathbf{\Phi}$). The generated CSI angles are organized in batches and used for training DNNs.

Data Preprocessing: As shown in Fig. 3, each batch of CSI angles are pre-processed before being used for training the DNNs. The pre-process is to make the angles zero-mean and unite-variance over all tones [36]. Albeit simple, this pre-process significantly improves the convergence of DNNs [36], especially when gradient descent algorithms are used for weight adaptation [37]. The AP also quantizes these pre-processed angles with different numbers of bits and keeps all versions to examine their performance.

Training Parameters and Provisions: Normalized mean squared error (NMSE) loss function is employed to measure the sparsification error. The DNNs are trained using Adam optimizer [38]. The training is performed with an initial learning rate of 0.001 and decaying rate of 0.98 following a step-wise approach. The batch size is set to 128. All parameters are initialized using Xavier initialization [39]. Dropout [40] is applied to all hidden layers to prevent over-fitting and improve the generalization of the model. All DNNs are trained end-to-end using Pytorch v1.4 library [41].

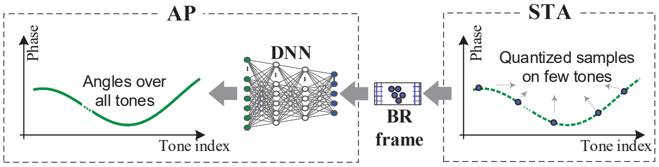


Fig. 6: DLCS workflow in sparsification (exploitation) phase.

C. Sparsification Phase

After completing the training phase, the AP initiates the sparsification phase. That is, the network begins to use the trained DNNs to reduce the channel sounding overhead when applicable. To do so, the AP informs all STAs of S_ψ, S_ϕ, q_ψ, and q_ϕ, where S_ψ and S_ϕ are the number of tones for which STAs report angles of $\mathbf{\Psi}$ and $\mathbf{\Phi}$, respectively. q_ψ and q_ϕ are the number of bits for quantizing each angle in $\mathbf{\Psi}$ and $\mathbf{\Phi}$, respectively. Fig. 6 illustrates the CSI reporting process when the AP is equipped with the trained DNNs. In what follows, we elaborate the operations at an STA and the AP, respectively.

Operations at an STA: Referring to Fig. 2, when MU-MIMO transmission is triggered by an NDPA frame, each STA estimates the downlink channel vector $\mathbf{H}(k)$ based on the received NDP frame, where $k = \{k_\psi, k_\phi\}$ is the selected tone indices, $k_\psi \in \{\lfloor 0.5N_{tone}/S_\psi \rfloor, \lfloor 1.5N_{tone}/S_\psi \rfloor, \cdots, \lfloor (S_\psi - 0.5)N_{tone}/S_\psi \rfloor\}$ is the set of tone indices for which STAs report $\mathbf{\Psi}$ and $k_\phi \in \{\lfloor 0.5N_{tone}/S_\phi \rfloor, \lfloor 1.5N_{tone}/S_\phi \rfloor, \cdots, \lfloor (S_\phi - 0.5)N_{tone}/S_\phi \rfloor\}$ is the set of tone indices for which STAs report $\mathbf{\Phi}$. Spatial compression is performed on $\mathbf{H}(k)$ to obtain the angles in $\mathbf{\Psi}$ and $\mathbf{\Phi}$, which are then quantized using q_ψ and q_ϕ bits (using the quantization method in [31]), respectively. In the BR frame shown in Fig. 2, instead of reporting CSI angles on all groups of tones, the STAs report ψ and ϕ angles only on those S_ψ tones and S_ϕ tones, respectively. In addition, each STA also reports the measured SNR values to the AP in the BR frame, following the existing 802.11 protocol [31].

Operations at the AP: Upon receiving the reports from an STA, the AP extracts the quantized angles and SNR reports. As illustrated in Fig. 6, the received angles are then fed into the DNNs to infer the angles over all tones. The output of the DNNs are then used to construct the beamforming vectors for downlink MU-MIMO transmissions.

D. Parameter Selection and Numerical Results

A question to ask is how to choose the values for sparsification parameters S_ψ, S_ϕ, q_ψ, and q_ϕ. In our design, the parameter values are selected to ensure the end-to-end

TABLE II: End-to-end error of DNNs in inferring the angles in $\mathbf{\Psi}$.

	S_ψ=5	S_ψ=6	S_ψ=7	S_ψ=8	S_ψ=9
q_ψ=3 bits	10.55%	10.63%	12.00%	8.99%	9.88%
q_ψ=4 bits	5.85%	4.95%	5.03%	3.86%	3.29%
q_ψ=5 bits	3.97%	2.77%	2.52%	1.93%	1.32%
q_ψ=6 bits	3.52%	2.16%	1.53%	1.35%	1.14%
q_ψ=7 bits	3.19%	2.08%	1.16%	1.14%	0.80%

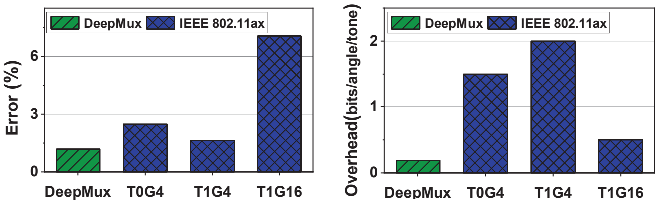
TABLE III: End-to-end error of DNNs in inferring the angles in $\mathbf{\Phi}$.

	S_ϕ=5	S_ϕ=6	S_ϕ=7	S_ϕ=8	S_ϕ=9
q_ϕ=3 bits	26.51%	22.70%	27.39%	29.83%	21.57%
q_ϕ=4 bits	8.30%	6.63%	6.33%	6.09%	5.73%
q_ϕ=5 bits	3.01%	2.40%	2.19%	2.14%	1.85%
q_ϕ=6 bits	2.67%	2.06%	1.10%	1.01%	0.76%
q_ϕ=7 bits	2.30%	1.07%	0.82%	0.77%	0.57%

errors below a pre-defined threshold, which is empirically set. Specifically, after the AP collects the sufficient channel data, it first trains DNNs under different values of sparsification parameters and then records the end-to-end error in the test phase. The AP selects the values for sparsification parameters that yield the lowest sounding overhead while meeting the end-to-end error requirement (below a pre-defined threshold).

To illustrate this selection approach, we resort to experiments. We implemented DLCS in an indoor environment and collected about $50,000$ angle samples in the uplink over 20 MHz bandwidth. We tuned those parameters and examined the performance of well-trained DNNs. As a possible end-to-end error threshold in inferring the angles, we use error from the tone grouping mechanism. Table II and Table III present our results. In each table, the DNN settings which meet the end-to-end error requirement are highlighted in green color. Based on the results, we choose $(S_\psi = 9, q_\psi = 5)$ which leads to 0.19 bits/angle/tone overhead and 1.32% error for the angles in $\mathbf{\Psi}$. We choose $(S_\phi = 6, q_\phi = 7)$ for the angles in $\mathbf{\Phi}$ which leads to 0.18 bits/angle/tone overhead and 1.07% error. Finally, the DNNs we choose are a $9 \times 16 \times 32 \times 64 \times 128 \times 234$ DNN for sparsification of $\mathbf{\Psi}$ and a $6 \times 16 \times 32 \times 64 \times 128 \times 234$ DNN for sparsification of $\mathbf{\Phi}$. We note that the resultant parameter values are scenario-specific. When an AP is moved to a new scenario, it needs to re-tune the parameters to obtain the "best" values for those parameters. Fortunately, the parameter re-tuning process can be done by the AP automatically without human intervention.

We now compare DLCS with existing 802.11 protocols in terms of error and sounding overhead. Fig. 7 presents our results. Particularly, TiGj in the figure means feedback type i is employed and $N_g = j$ tones are grouped for feedback. Fig. 7(a) shows the superior performance of DLCS in terms of error. DLCS reaches 1.19% error, while T0G4, T1G4, and T1G16 reach 2.48%, 1.64%, and 7.05% error, respectively. Fig. 7(b) shows that DLCS entails significantly lower overhead compared to existing 802.11 protocols. DLCS reaches a sounding overhead as low as 0.19 bits/angles/tone while T0G4, T1G4, and T1G16 reach 1.50, 2.00, and 0.50 bits/angles/tone overhead, respectively. This means DLCS reduces sounding overhead by $62.0\%{\sim}90.5\%$.



(a) Error.

(b) Overhead.

Fig. 7: Error and overhead comparison between DLCS and existing 802.11 protocols [29].

VI. DLRA: A LIGHTWEIGHT RESOURCE ALLOCATION

In this section, we employ DNNs to facilitate the resource allocation problem at the AP, which includes two subproblems: RU assignment and power allocation. Recall that the AP recovers angles in $\mathbf{\Psi}$ and $\mathbf{\Phi}$ using DNNs, and it also collects SNR values over all tones. The angles in $\mathbf{\Psi}$ and $\mathbf{\Phi}$ can be used to partially reconstruct the right singular vectors of channel matrices, which can be leveraged to mitigate inter-user interference in the downlink transmissions. The SNR values provide the information of channel quality, which can be used to optimize the resource allocation. In what follows, we first formulate the resource allocation problem as an optimization problem, and then develop a learning-based algorithm to solve it. Finally, we offer numerical results to show the effectiveness of the proposed learning-based algorithm.

A. Problem Formulation and Reformulation

Problem Formulation: At an AP, denote $\mathcal{N}$ as the set of STAs that it serves in the downlink MU-MIMO-OFDMA transmission. Denote $\mathcal{R}$ as the set of RUs, which are the granularity for assignment. Let $|\mathcal{N}| = N_{\text{sta}}$ and $|\mathcal{R}| = N_{\text{ru}}$. We define a binary variable $z_{i,j}$ to indicate the RU assignment. Specifically, $z_{i,j} = 1$ if RU j is assigned to STA i; and $z_{i,j} = 0$ otherwise. Denote $p_{i,j}$ as the portion of the AP's power allocated to STA i on RU j. Denote W_j as the bandwidth of RU j. Denote $\gamma_{i,j}$ as reported SNR at STA i on RU j. Denote $r_{i,j}$ as the data rate achieved by STA i on RU j. Denote r_i as the achievable data rate for STA i. Denote $\Omega(\cdot)$ as the mapping function from SNR to data rate.

Then, the resource allocation problem with the objective of maximizing total STAs' data rate can be expressed as:

$$\underset{\underline{p},\underline{z}}{\text{maximize}} \sum_{i\in\mathcal{N}} r_i \tag{2a}$$

$$\text{s.t.} \quad r_i \leq \sum_{j\in\mathcal{R}} r_{i,j}, \quad i\in\mathcal{N}; \tag{2b}$$

$$r_{i,j} \leq W_j z_{i,j} \Omega\left(p_{i,j}\gamma_{i,j}\right), \quad i\in\mathcal{N}, j\in\mathcal{R}; \tag{2c}$$

$$\sum_{i\in\mathcal{N}} z_{i,j} \leq N_{\text{ap}}, \quad j\in\mathcal{R}; \tag{2d}$$

$$\sum_{i\in\mathcal{N}, j\in\mathcal{R}} p_{i,j} \leq 1\,. \tag{2e}$$

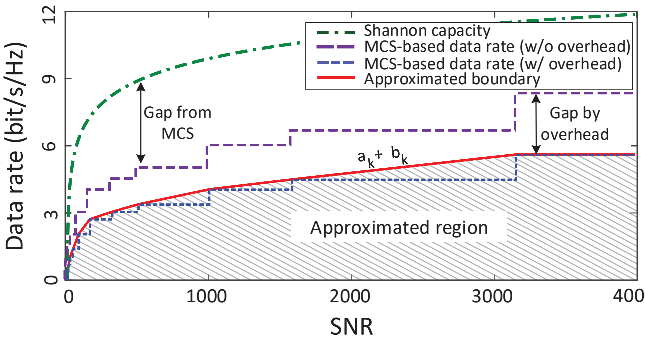


Fig. 8: Illustration of Shannon capacity, MCS-based data rate, and achievable data rate.

In this formulation, $\underline{z} = \{z_{i,j}\}_{i\in\mathcal{N},j\in\mathcal{R}}$ and $\underline{p} = \{p_{i,j}\}_{i\in\mathcal{N},j\in\mathcal{R}}$ are optimization variables. $\{\gamma_{i,j}\}_{i\in\mathcal{N},j\in\mathcal{R}}$, $\{W_j\}_{j\in\mathcal{R}}$, and N_{ap} are given parameters. Constraint (2b) calculates the achieved data rate by an STA over all RUs. Constraint (2c) defines the achievable rate region. Constraint (2d) is spatial DoF constraints on the maximum number of STAs that can be allocated to an RU. Constraint (2e) characterizes the power budget at the AP.

Achievable Rate Region: A classical way to map SNR to data rate is Shannon capacity. However, Shannon capacity is a theoretical bound and hard to reach in practice. In 802.11 networks, adaptive MCS (modulation and coding scheme) is used to adjust the data rate based on SNR. As shown in Fig. 8, there is a significant gap between Shannon capacity and MCS-based data rate. Therefore, Shannon capacity is not an ideal function for our purpose. Moreover, when taking into account the overhead from OFDM cyclic prefix and pilot tones in 802.11ax[2], the achievable data rate becomes even lower, as shown in Fig. 8. The achievable data rate region (MCS-based rate with overhead) is characterized by a staircase curve, which is non-convex function. To simplify the optimization, we approximate the achievable rate region using a series of linear constraints as illustrated by Fig. 8.

Mathematically, by defining γ as a measured SNR value, we approximate the achievable rate region as follows:

$$\Omega\left(\gamma\right) \leq a_k\gamma + b_k; \qquad k \in \mathcal{K}, \tag{3}$$

where a_k and b_k are given in Table IV as per IEEE 802.11ax; and $\mathcal{K} \triangleq \{1, 2, \cdots, 13\}$. We note that the EVM in Table IV is equivalent to the inverse of post-SNR of a decoded data stream at a receiver. The relation of γ in (3) and the EVM value in Table IV can be expressed as $\gamma = 10^{-\text{EVM}/10}$.

Based on the EVM regions specified in Table IV, the approximated achievable rate region with its boundaries is shown in Fig. 8. Then, constraints in (2c) can be expressed as:

$$r_{i,j} \leq W_j z_{i,j}(a_k p_{i,j}\gamma_{i,j} + b_k), \quad i \in \mathcal{N}, j \in \mathcal{R}, k \in \mathcal{K}. \tag{4}$$

TABLE IV: EVM specified in IEEE 802.11ax standard [29].

EVM (dB)	$[+\infty,-5)$	$[-5,-8)$	$[-8,-10)$	$[-10,-13)$	$[-13,-16)$	$[-16,-19)$	$[-19,-22)$
Modulation	N/A	BPSK	BPSK	QPSK	QPSK	16QAM	16QAM
Coding rate	N/A	1/2	3/4	1/2	3/4	1/2	3/4
Γ(EVM)	N/A	1/2	3/4	1	3/2	2	3
a_i	0.1067	0.0536	0.0457	0.0339	0.0170	0.0170	0.0085
b_i	0	0.1679	0.2177	0.3359	0.6734	0.6718	1.3468
EVM (dB)	$[-22,-25)$	$[-25,-27)$	$[-27,-30)$	$[-30,-32)$	$[-32,-35)$	$[-35,-\infty)$	$[-35,-\infty)$
Modulation	64QAM	64QAM	64QAM	256QAM	256QAM	1024QAM	1024QAM
Coding rate	2/3	3/4	5/6	3/4	5/6	3/4	5/6
Γ(EVM)	4	9/2	5	6	20/3	15/2	25/3
a_i	0.0021	0.0018	0.0013	0.0008	0.0007	N/A	0
b_i	2.3609	2.4605	2.6968	3.2806	3.3696	N/A	5.6250

Using (4), the resource allocation problem in (2) can be re-defined as:

$$\begin{aligned} &\underset{\underline{p},\underline{z}}{\text{maximize}} \sum_{i\in\mathcal{N}} r_i \\ &\text{s.t.} \quad \text{(2b), (2d), (2e), and (4).} \end{aligned} \tag{5}$$

The optimization problem in (5) is an MINLP problem. The non-linear term is from (4), where binary and continuous optimization variables are multiplied.

Problem Reformulation: To reduce the processing time, we reformulate the MINLP problem (5) to an MILP problem by leveraging a classic linearization technique [42]. To do so, we assume that the SNR value is bounded. This is a valid assumption in practice. Denote γ_{max} as the maximum value of SNR (e.g., 45 dB in our design) and define a constant $A = \max_{j,k}\{W_j(a_k\gamma_{max}+b_k)\}$. Then, (4) can be equivalently expressed as:

$$r_{i,j} \leq W_j(a_k p_{i,j}\gamma_{i,j} + b_k), \quad i \in \mathcal{N}, j \in \mathcal{R}, k \in \mathcal{K}. \tag{6a}$$

$$0 \leq r_{i,j} \leq z_{i,j}A, \quad i \in \mathcal{N}, j \in \mathcal{R}. \tag{6b}$$

Therefore, the MINLP problem in (5) can be reformulated to the following MILP problem:

$$\begin{aligned} &\underset{\underline{p},\underline{z}}{\text{maximize}} \sum_{i\in\mathcal{N}} r_i \\ &\text{s.t.} \quad \text{(2b), (2d), (2e), and (6).} \end{aligned} \tag{7}$$

We note that the MINLP problem in (5) and the MILP problem in (6) have identical feasible region. The reformulation does not alter the solution space. The new optimization problem involves $2N_{\text{sta}}N_{\text{ru}} + N_{\text{sta}}$ continuous variables, $N_{\text{sta}}N_{\text{ru}}$ binary variables, and $14N_{\text{sta}}N_{\text{ru}} + N_{\text{sta}} + N_{\text{ru}} + 1$ constraints. Recall the example in Section III-B, where a 4-antenna AP serves six STAs on four 52-tone RUs. By formulating the resource allocation problem in the form of (7), off-the-shelf optimization solver MOSEK [43] can find an optimal solution within 5 seconds for most cases. In general, MILP is NP-hard. Its computational complexity is still beyond the acceptable range of a wireless AP device.

B. DLRA: A Deep-Learning-Based Resource Allocation

Solving an MILP problem is still beyond the computational capacity of an 802.11ax-enabled AP to allocate its resources for downlink transmissions. To reduce the computational complexity, we take advantage of recent advances in DNNs. Specifically, we first reformulate the resource allocation problem as an MILP problem as shown in (7), and then employ a DNN to compute the binary variables. Once the binary

[2] For 802.11ax with 20 MHz bandwidth, every 26-tone RU has 2 tones for pilot.

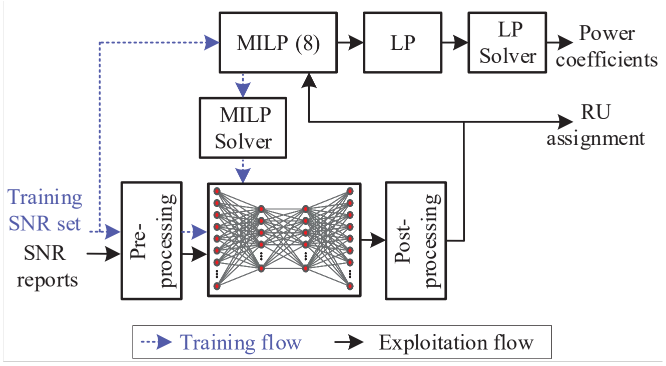


Fig. 9: DLRA workflow in training and exploitation phases.

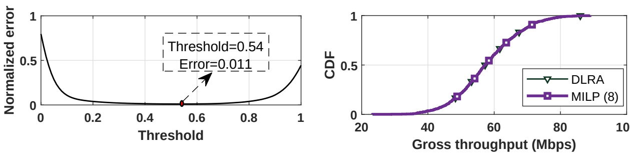
variables are determined, the MILP problem degrades to a linear programming problem, which is easy to solve. In what follows, we focus on the design of a DNN to determine the binary variables in (7).

DNN Settings: Fig. 9 shows the DNN-based approach in training and sparsification (exploitation) phases. The input of the DNN is the SNR values reported by the STAs. The dimension of input layer is $N_{\rm sta}N_{\rm ru}$. The DNN consists of multiple hidden layers. Each hidden layer is fully-connected, followed by a batch-normalization layer to speed up the training convergence [35]. Sigmoid activation function is used for each layer. The output layer has $N_{\rm sta}N_{\rm ru}$ neurons, each of which corresponds to a binary variable in RU assignment sub-problem. In our experiments, we consider the case where an 8-antenna AP serves 20 STAs on 9 RUs. For this case, the input and output layers both have 180 neurons, and the overall DNN's structure we trained for RU assignment is $180 \times 128 \times 128 \times 180$.

Data Collection and Pre-processing: We collect $60,000$ SNR reports from an office environment. Each report consists SNR values over all the nine 26-tone RUs on 20 MHz bandwidth. Every set of SNR values (20 SNR reports) will be flattened, normalized, and then used for training the DNN as an instance of its input. At the same time, the set of unprocessed SNR values will be fed into (7). The output of (7) includes RU assignment and power allocation coefficients. The resultant RU assignment will be used as the reference output of the DNN in its supervised training procedure. We use MOSEk v.9 [43] to solve (7) for a given set of SNR values. Since data generation process is pretty slow, we augment the training data set by adding negligible noise to the original input samples. Moreover, we set aside one third of input-output sample pairs for test purpose. We augment the remaining samples 4.5 times.

Training Process: To train the DNN, we use NMSE loss function. The outputs of (7) for given sets of SNR values are used as reference outputs of the DNN in training loss calculation. For training the DNN, we use Adam optimizer [38] and PyTorch v1.4 library [41]. We also apply batch normalization [35] and Xavier initialization [39] approaches to accelerate the training process.

Post-Processing: The output of DNN will be post-processed in two steps: *binarization* and *correction*. The output of DNN is a vector comprising real values bounded between 0 and 1. We apply a threshold-based *binarization* on outputs of the DNN to transform them into binary entities. Once the binary vector is obtained, we can use our *domain knowledge* to further polish this vector. Two rules are followed in the *correction* step: i) If the DoF constraint is violated on an RU, the STA with the lowest SNR will be removed until the DoF constraint is met. ii) When the DoFs on an RU are under-utilized, the STA with the highest SNR will be activated if there is an assigned STA with a lower SNR.



(a) Binarization threshold versus normalized error.

(b) Performance gap between DLRA and the optimum to (7).

Fig. 10: Illustrating the performance of DLRA when compared to an optimal solution.

Computational Complexity: Referring to Fig. 9, the computational complexity of pre-processing and post-processing operations is $\mathcal{O}\left(N_{\rm sta}\right)$, provided that $N_{\rm ru}{<}N_{\rm sta}$. For the trained DNN, assuming that the size of hidden layer is proportional to the size of input, its computational complexity is $\mathcal{O}\left(N_{\rm sta}^2\right)$. For a given RU assignment, MILP in (5) degrades to an LP problem. The computational complexity of solving the LP problem is $\mathcal{O}\left(N_{\rm sta}^{2.5}\right)$. Therefore, the overall complexity of DLRA is $\mathcal{O}\left(N_{\rm sta}^{2.5}\right)$.

Numerical Results: After the DNN is trained, we use a set of data samples to test its performance. We examine the accuracy of DNN output when different thresholds are used for the binarization post-processing. Fig. 10(a) shows the results. It can be seen, DLRA reaches 98.9% accuracy when using 0.54 as the binarization threshold. This means that DLRA offers a very accurate RU assignment. We measured the performance gap between two cases, where the AP uses DLRA and where the AP uses MILP problem for resource allocation. As shown in Fig. 10(b), the results confirm that the DLRA almost reaches the optimal performance.

VII. Experimental Evaluation

In this section, we evaluate the performance of DeepMux by comparing it with existing 802.11ax protocols.

A. *Experimental Settings*

Wireless Testbed and Experimental Setting: Fig. 11(a) and Fig. 11(b) show the wireless testbed that we use to evaluate DeepMux. The testbed has one AP and four STAs which are built using USRP N210 devices and general computers. The AP is equipped with 8 antennas while each STA is equipped with one antenna. As shown in Fig. 11(c), the AP is placed at a fixed location, while the four STAs have many random locations to be placed.

Implementation of 802.11ax: The 802.11ax protocol in Fig. 2 is implemented on the testbed. The carrier frequency is set to 2.484 GHz, and the bandwidth is set to 20 MHz. Due to

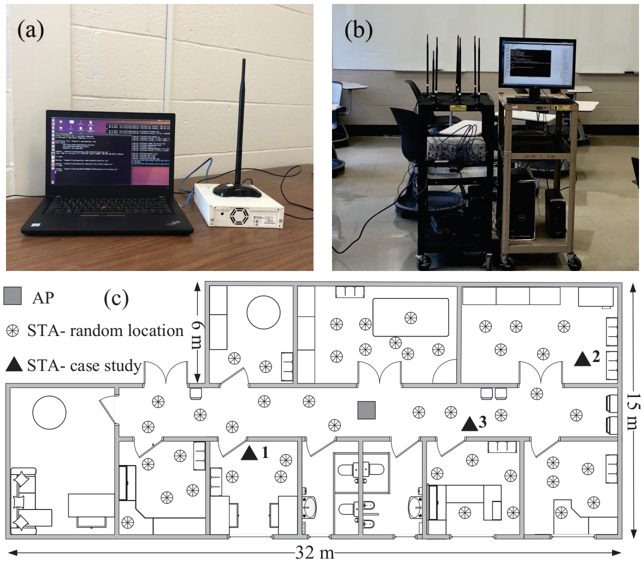


Fig. 11: Illustrating our wireless testbed and test environment. (a) Prototyped STA. (b) Prototyped AP. (c) Floor plan of tests.

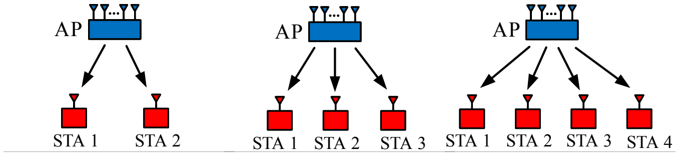
the hardware limitation, the inter-frame spacing is equal to one second. A frame has 256 tones in its OFDM modulation, with 18 pilot tones, 216 payload tones, and 22 unused tones. The 26-tone RU configuration (see Fig. 1) is used in our study. The transmission power of the AP and STAs is set to 15 dBm. The signal processing modules at both AP and STAs are implemented using C++ in GNURadio-Companion. LDPC channel encoding and decoding are not implemented to reduce the implementation complexity.

Implementation of DeepMux: DeepMux is implemented on top of the 802.11ax protocol, and its DNNs are trained at the AP using Pytorch v1.4 library [41]. To train DNNs, our data collection campaign lasted three days. During the campaign, low and moderate human activities (i.e., $0{\sim}5$ persons with brisk walking speed) were observed in the environment shown in Fig. 11(c). In this campaign, $100,000$ angles ($50,000$ vectors in $\mathbf{\Psi}$ and $50,000$ vectors in $\mathbf{\Phi}$) on 234 tones were collected for DLCS to train its two DNNs. Meanwhile, $60,000$ SNR reports were collected from the BR frames for DLRA to train its DNN.

B. Performance Metrics

Error Vector Magnitude (EVM): EVM is widely used to measure the quality of received signal. Mathematically, EVM is defined as: $\mathrm{EVM} = 10\log_{10}\left(\frac{\mathbb{E}[|\hat{X}-X|^2]}{\mathbb{E}[|X|^2]}\right)$, where X and $\hat{X}$ are original and estimated signals, respectively.

Gross Throughput: Gross throughput is the over-the-air data rate achieved by an STA or the AP. It can be inferred based on the measured EVM by $r = \frac{N_\mathrm{p}}{N_\mathrm{fft}+N_\mathrm{cp}} \cdot b \cdot \Gamma\left(\mathrm{EVM}\right)$, where r is the gross throughput, N_p is the number of payload tones, N_fft is FFT points, N_cp is the length of cyclic prefix, b is the sampling rate, and $\Gamma(\mathrm{EVM})$ is the average number of bits carried by one tone, as specified in Table IV. $\Gamma(\mathrm{EVM})$ is determined by modulation order and (LDPC) coding rate.



(a) Two-user case. (b) Three-user case. (c) Four-user case.

Fig. 12: Test scenarios used for evaluation of DLCS.

Net Throughput: Net throughput calculates the data rate while taking into account channel sounding airtime overhead. It can be expressed as: $r_\mathrm{net} = \frac{t_{payload}}{t_{payload}+t_\mathrm{overhead}} \cdot r$, where $t_{payload}$ and t_overhead are the time duration of data transmission and channel sounding, respectively. t_overhead is determined by the airtime used for transmitting BR, NDPA, NDP, and TBRP frames. For simplicity, we do not consider inter-frame space, re-transmission, and frame aggregation in our calculations.

Comparison Baselines: For DLCS, we compare it with the tone grouping approaches specified in 802.11ax. For notational simplicity, we use TiGj to denote the 802.11 channel sounding protocol with feedback type $i \in \{0, 1\}$ and $j \in \{4, 16\}$ tones in each group. For DLRA, there is not a standardized baseline for comparison. Hence, we implement the best resource allocation effort onto IEEE 802.11ax. The best effort is full utilization of available DoFs on each RU.

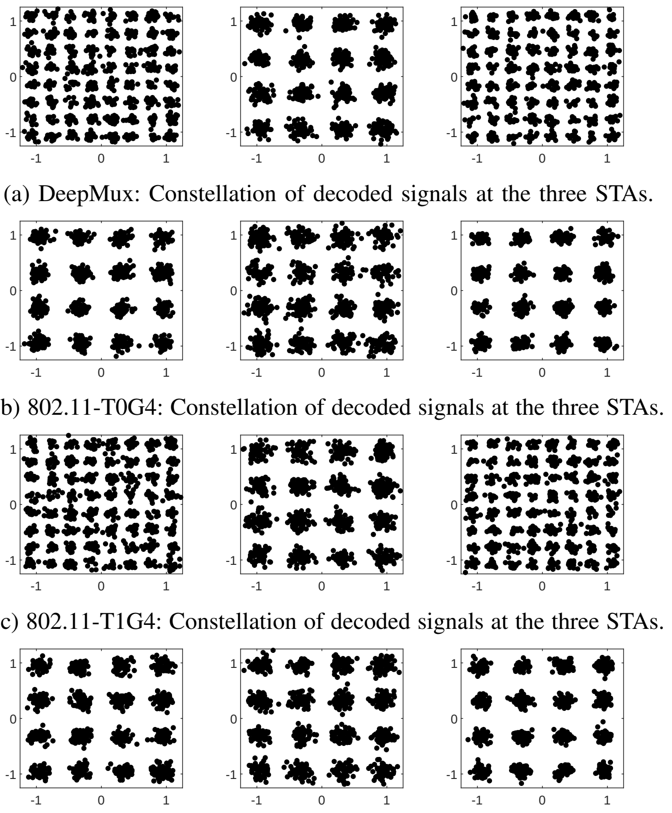
C. A Case Study for DLCS

We consider the case as shown in Fig. 12(b), where the AP serves three STAs. The AP is placed at the square mark in Fig. 11(c), and the three STAs are placed at the triangle marks in the figure. Every RU serves these three STAs with equal power allocation, and no resource allocation is involved in this study. In what follows, we present our results.

Constellation: We perform downlink MU-MIMO transmissions using both 802.11ax and DLCS channel sounding protocols and collect the decoded signals at the three STAs. Fig. 13 shows the constellations of decoded signals at the three STAs. The EVMs of the decoded signals are presented in Table V. It can be seen from the measured EVMs that DeepMux offers the best signal quality in the downlink transmissions. This is because the DNNs at the AP can accurately recover CSI over all tones based on the limited CSI feedback. It also can be seen from Fig. 13 that DeepMux and 802.11-T1G4 achieve similar signal quality (constellation) in the downlink. This is because we used 802.11-T1G4 as the performance benchmark to select the DNN parameters for DLCS in our experiments.

Feedback Overhead: DeepMux entails 0.6 kbit overhead for CSI feedback from each STA. In contrast, 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 entails 4.9 kbit, 6.5 kbit, and 1.6 kbit overhead for CSI feedback, respectively.

EVM, Gross Throughput, and Net Throughput: Table V presents our experimental results. We have the following observations. First, in terms of EVM and gross throughput, DLCS is slightly better than 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16. Second, in terms of net throughput, DLCS is significantly superior to 802.11-T0G4, 802.11-T1G4, and



(a) DeepMux: Constellation of decoded signals at the three STAs.

(b) 802.11-T0G4: Constellation of decoded signals at the three STAs.

(c) 802.11-T1G4: Constellation of decoded signals at the three STAs.

(d) 802.11-T1G16: Constellation of decoded signals at the three STAs.

Fig. 13: Constellations of decoded signals at STA 1 (left), STA 2 (middle), and STA 3 (right), when the WLAN uses different feedback protocols.

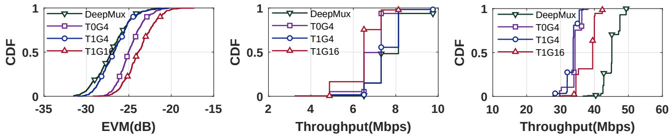
TABLE V: A case study for comparing DLCS of DeepMux with 802.11 protocols.

		STA 1	STA 2	STA 3	AP
DeepMux	EVM (dB)	-23.5	-19.1	-24.0	–
	Per RU Gross throughput (Mbps)	6.5	4.9	6.5	17.9
	Net throughput (Mbps)	19.1	14.3	19.1	52.5
T0G4	EVM (dB)	-20.1	-17.6	-21.3	–
	Per RU Gross throughput (Mbps)	4.9	3.2	4.9	13.0
	Net throughput (Mbps)	13.7	9.1	13.7	36.5
T1G4	EVM (dB)	-23.0	-17.9	-23.6	–
	Per RU Gross throughput (Mbps)	4.9	3.2	4.9	13.0
	Net throughput (Mbps)	14.7	7.4	14.7	36.8
T1G16	EVM (dB)	-19.8	-18.2	-20.7	–
	Per RU Gross throughput (Mbps)	6.5	3.2	6.5	16.2
	Net throughput (Mbps)	15.6	10.4	15.6	41.6

802.11-T1G16. This is not surprising because DLCS consumes much lower airtime for CSI feedback compared to 802.11 channel sounding protocols.

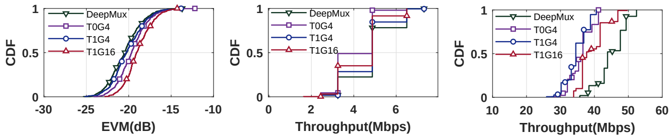
D. Extensive Results of DLCS

We extend the case study to extensive experimental trials to thoroughly examine the performance of DLCS. We consider three cases: two-user, three-user, and four-user MIMO as shown in Fig. 12. The AP serves these two/three/four users exclusively on all RUs, with equal power allocation. Each STA is placed at a randomly selected spot marked with a filled circle in Fig. 11(c).



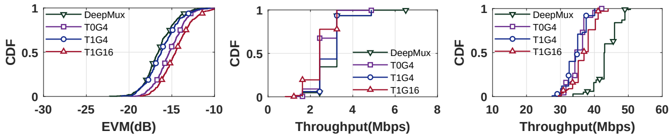
(a) EVM (b) Gross throughput (c) Net throughput

Fig. 14: Comparison of DeepMux and 802.11 protocols in two-user MIMO downlink transmission.



(a) EVM (b) Gross throughput (c) Net throughput

Fig. 15: Comparison of DeepMux and 802.11 protocols in three-user MIMO downlink transmission.



(a) EVM (b) Gross throughput (c) Net throughput

Fig. 16: Comparison of DeepMux and 802.11 protocols in four-user MIMO downlink transmission.

Two-User Case: Fig. 14 presents the comparison results of DeepMux and 802.11 protocols in terms of EVM, gross throughput, and net throughput. Per Fig. 14(a), DeepMux achieves -27.1 dB EVM on average, while 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 reach -24.7 dB, -26.7 dB, and -23.8 dB EVM, respectively. Per Fig. 14(b), DeepMux slightly outperforms 802.11 protocols in terms of gross throughput. DeepMux achieves 7.7 Mbps gross throughput per RU on average. In contrast, 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 achieve 6.8 Mbps, 7.5 Mbps, and 6.4 Mbps gross throughput per 26-tone RU, respectively.

Net throughput reflects the advantage of DLCS as it takes into account airtime overhead in the calculation of throughput. As shown in Fig. 14(c), DeepMux obtains 45.2 Mbps net throughput on all RUs on average. In contrast, 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 achieve 34.2 Mbps, 33.5 Mbps, and 38.2 Mbps net throughput, respectively. DeepMux offers 31.6%, 34.3%, and 17.8% net throughput gains compared to 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16, respectively.

Three-User Case: The observations in three-user case are consistent with those in two-user case. Fig. 15 shows the experimental results. DeepMux slightly outperforms 802.11 protocols in terms of EVM and gross throughput. Per Fig. 15(a), DeepMux achieves -20.4 dB EVM on average, while 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 achieve -19.6 dB, -20.1 dB, and -18.9 dB EVM, respectively. Per Fig. 15(b),

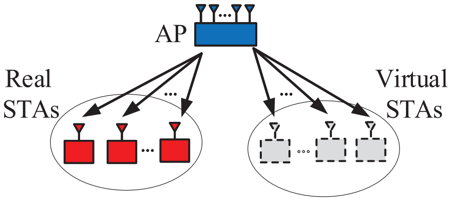


Fig. 17: Test scenario for evaluating DeepMux in MU-MIMO-OFDMA transmissions.

DeepMux achieves 4.9 Mbps gross throughput on average per RU, while 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 achieve 4.1 Mbps, 4.6 Mbps, and 4.4 Mbps respectively. DeepMux offers a significant gain of net throughput over 802.11 protocols. Per Fig. 15(c), DeepMux obtains 45.2 Mbps net throughput on average. In contrast, 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 achieve 36.1 Mbps, 34.8 Mbps, and 39.4 Mbps net throughput, respectively. This indicates that DeepMux offers 25.2%, 30.0%, and 14.7% gains compared to 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16, respectively.

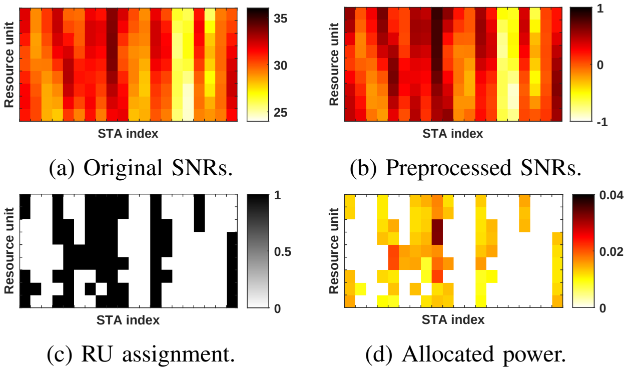
Four-User Case: The observations in this case are consistent with those in previous two cases. Fig. 16 presents the experimental results. In the end, DeepMux achieves 43.7 Mbps net throughput on average. In contrast, 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 achieve 35.2 Mbps, 34.6 Mbps, and 37.0 Mbps net throughput, respectively. Numerically, DeepMux offers 24.1%, 26.3%, and 18.1% net throughput gains compared to 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16, respectively.

E. *Overall Performance of DeepMux*

Methodology: The full evaluation of DeepMux requires a large-scale wireless testbed with many STAs to mimic real 802.11ax networks in MU-MIMO-OFDMA transmissions. However, we do not have such a luxury. We therefore use a hybrid approach that combines emulation and experimentation to evaluate DeepMux. Fig. 17 shows our testbed setting, where the AP serves 4 real STAs and 16 virtual STAs. The 4 real STAs perform over-the-air transmissions, while the 16 virtual STAs are created by the AP based on the pre-stored CSI from other locations. The virtual STAs are used for DLRA. In the downlink transmission, the AP sends precoded signals to all (real and virtual) STAs, and the performance is measured at STAs.

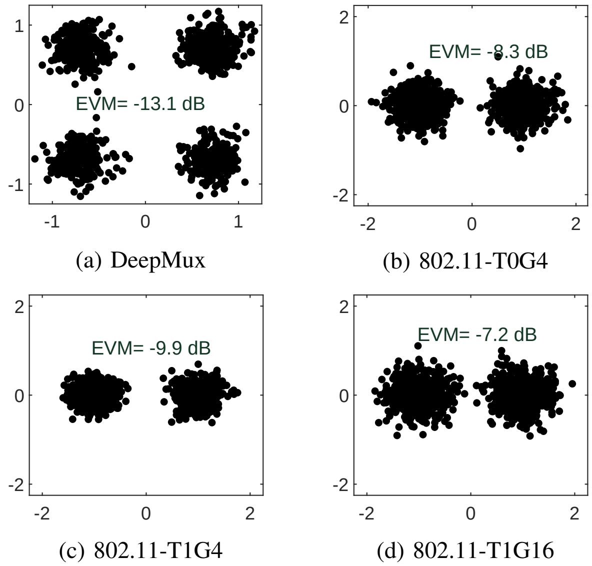
A Close Look into DLRA: As a case study, we place one of real STAs at the locations marked by triangle 1 in Fig. 11(c). Fig. 18(a) shows the SNR values from the real and virtual STAs. The reported SNR values are first preprocessed for normalization, as shown in Fig. 18(b). The normalized values are then fed into a DNN for RU assignment. Fig. 18(c) shows the RU assignment results from the DNN. With the RU assignment results from DNN, the optimization problem in (7) degrades to an LP problem. The LP problem is then solved to obtain the power allocation results, which are shown in Fig. 18(d).

Referring to Fig. 18(d), the rightmost column denotes RU assignment and allocated power to the STA of interest. Fig. 19 shows the constellation of received signal by the mentioned STA on the first RU with the aid of DeepMux and existing protocols in 802.11ax. The results reveal superior performance of DeepMux in terms of EVM. For this STA, the gross throughput achieved on the first RU is 2.4 Mbps, 1.2 Mbps, 1.2 Mbps, and 0.8 Mbps with DeepMux, 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16, respectively. The net throughput achieved by this user on the first RU is 9.1 Mbps, 4.9 Mbps, 6.5 Mbps, and 4.7 Mbps with DeepMux, 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16, respectively. Over all RUs, DeepMux obtains 43.5 Mbps net throughput, while 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 respectively achieve 31.7 Mbps, 29.9 Mbps, and 37.8 Mbps.



(a) Original SNRs.
(b) Preprocessed SNRs.
(c) RU assignment.
(d) Allocated power.

Fig. 18: A case study on resource allocation by DLRA.

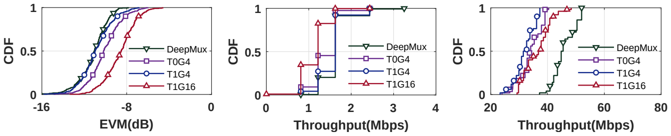


(a) DeepMux
(b) 802.11-T0G4
(c) 802.11-T1G4
(d) 802.11-T1G16

Fig. 19: EVM of decoded signal on first STA over first RU.

Extensive Results: To obtain more comprehensive results, we place the four real STAs at different locations marked with filled circles in Fig. 11(c). The experimental results are summarized as follows.

- *EVM:* Fig. 20(a) presents the measured EVM at STAs. On average, DeepMux achieves −11.2 dB EVM for STAs, while 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 reach −10.1 dB, −10.9 dB, and −8.6 dB EVM, respectively.



(a) EVM (b) Gross throughput (c) Net throughput

Fig. 20: Comparison of DeepMux and 802.11 protocols when an 8-antenna AP serves 20 stations on 20 MHz bandwidth.

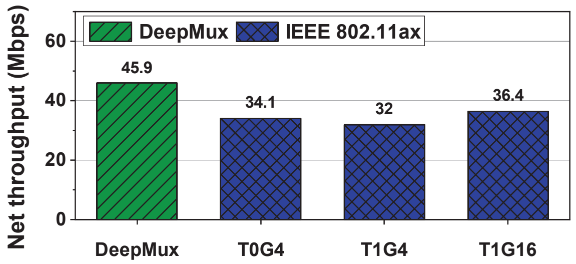


Fig. 21: Average net throughput achieved by DeepMux and 802.11 protocols.

- *Gross Throughput per RU:* Fig. 20(b) presents the gross throughput per RU. Specifically, DeepMux achieves 1.6 Mbps gross throughput per 26-tone RU. In contrast, 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 achieve 1.4 Mbps, 1.6 Mbps, and 1.1 Mbps, respectively.
- *Net Throughput:* Fig. 20(c) shows the net throughput achieved by different protocols, and Fig. 21 shows the average net throughput at the AP. Specifically, DeepMux achieves 45.9 Mbps net throughput on average. In contrast, 802.11-T0G4, 802.11-T1G4, and 802.11-T1G16 achieve 35.7 Mbps, 32.0 Mbps, and 36.4 Mbps, respectively. The net throughput gain of DeepMux is 34.9% compared to 802.11-T0G4, 43.6%, compared to 802.11-T1G4, and 26.3% compared to 802.11-T1G16.

VIII. Conclusion

In this paper, we presented DeepMux, a deep-learning-based approach to enhance the efficiency of downlink MU-MIMO-OFDMA transmissions in 802.11ax networks. DeepMux is designed upon two components, namely DLCS and DLRA, both of which reside in APs and impose no computation/communication burden to Wi-Fi clients. DLCS leverages DNNs to reduce overhead of CSI feedback in 802.11 protocols. It uses uplink channels to train the DNNs for downlink channels, making the training process easy to implement. Numerical results show that it can reduce the sounding overhead by 62.0%∼90.5% without sacrificing CSI feedback accuracy. DLRA tackles an MILP resource allocation problem by decoupling its integer and continuous optimization subproblems and employing a DNN to compute a solution to the integer part. Numerical results show that DLRA can achieve 98.9% optimality in RU assignment while bearing a low computational complexity. We have built a wireless testbed to examine the performance of DeepMux in an indoor environment. Experimental results show that DeepMux increases network throughput by 26.3%∼43.6% compared to 802.11 protocols.

References

[1] Wi-Fi® Alliance, “Wi-Fi® in 2019.” *Wi-Fi Alliance*, 21-Feb-2019. [Online]. Available: https://www.wi-fi.org/news-events/newsroom/wi-fi-in-2019, [Accessed: 21-Nov-2020].

[2] O. Bejarano, E. Magistretti, O. Gurewitz, and E. W. Knightly, “MUTE: Sounding inhibition for MU-MIMO WLANs,” in *Proc. 11th Annu. IEEE Int. Conf. Sens., Commun., Netw. (SECON)*, pp. 135–143, 2014.

[3] X. Ma, Q. Gao, J. Wang, V. Marojevic, and J. H. Reed, “Dynamic sounding for multi-user MIMO in wireless LANs,” *IEEE Trans. Consum. Electron.*, vol. 63, no. 2, pp. 135–144, 2017.

[4] J. Choi, S. Choi, and K. B. Lee, “Sounding node set and sounding interval determination for IEEE 802.11 ac MU-MIMO,” *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 10069–10074, 2016.

[5] R. E. Guerra, N. Anand, C. Shepard, and E. W. Knightly, “Opportunistic channel estimation for implicit 802.11 af MU-MIMO,” in *Proc. 28th Int. Teletraffic Congr. (ITC)*, vol. 1, pp. 60–68, 2016.

[6] H. Lou, M. Ghosh, P. Xia, and R. Olesen, “A comparison of implicit and explicit channel feedback methods for MU-MIMO WLAN systems,” in *Proc. IEEE 24th Annu. Int. Symp. Personal Indoor Mobile Radio Commun. (PIMRC)*, pp. 419–424, 2013.

[7] X. Xie, X. Zhang, and K. Sundaresan, “Adaptive feedback compression for MIMO networks,” in *Proc. of ACM MobiCom*, pp. 477–488, 2013.

[8] P. Kheirkhah Sangdeh, H. Pirayesh, A. Mobiny, and H. Zeng, “LB-SciFi: Online learning-based channel feedback for MU-MIMO in wireless LANs,” in *Proc. IEEE Int. Conf. Netw. Protocols (ICNP)*, pp. 1–11, 2020.

[9] X. Li and H. Wu, “Spatio-temporal representation with deep neural recurrent network in MIMO CSI feedback,” *IEEE Wireless Commun. Lett.*, 2020.

[10] T. Wang, C.-K. Wen, S. Jin, and G. Y. Li, “Deep learning-based CSI feedback approach for time-varying massive MIMO channels,” *IEEE Wireless Commun. Lett.*, vol. 8, no. 2, pp. 416–419, 2018.

[11] C. Lu, W. Xu, H. Shen, J. Zhu, and K. Wang, “MIMO channel information feedback using deep recurrent network,” *IEEE Commun. Lett.*, vol. 23, no. 1, pp. 188–191, 2019.

[12] C.-K. Wen, W.-T. Shih, and S. Jin, “Deep learning for massive MIMO CSI feedback,” *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 748–751, 2018.

[13] J. Guo, C.-K. Wen, S. Jin, and G. Y. Li, “Convolutional neural network based multiple-rate compressive sensing for massive MIMO CSI feedback: Design, simulation, and analysis,” *IEEE Trans. Wireless Commun.*, 2020.

[14] D. Weller, R. D. Mensenkamp, A. van der Vegt, J.-W. van Bloem, and C. de Laat, “Wi-Fi 6 performance measurements of 1024-QAM and DL OFDMA,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, pp. 1–7, 2020.

[15] K. Dovelos and B. Bellalta, “A scheduling policy for downlink OFDMA in ieee 802.11 ax with throughput constraints,” *arXiv preprint arXiv:2009.00413*, 2020.

[16] M. Ş. Kuran, A. Dilmac, Ö. Topal, B. Yamansavascilar, S. Avallone, and T. Tugcu, “Throughput-maximizing OFDMA scheduler for IEEE 802.11 ax networks,” in *Proc. IEEE 31st Annu. Int. Symp. Personal Indoor Mobile Radio Commun. (PIMRC)*, pp. 1–7.

[17] K. Wang and K. Psounis, “Scheduling and resource allocation in 802.11 ax,” in *Proc. IEEE INFOCOM*, pp. 279–287, 2018.

[18] K. Wang and K. Psounis, “Efficient scheduling and resource allocation in 802.11 ax multi-user transmissions,” *Comput. Commun.*, vol. 152, pp. 171–186, 2020.

[19] M. Inamullah, B. Raman, and N. Akhtar, “Will my packet reach on time? deadline-based uplink OFDMA scheduling in 802.11 ax WLANs,” in *Proc. ACM MSWiM*, pp. 181–189, 2020.

[20] D. Bankov, A. Didenko, E. Khorov, and A. Lyakhov, “OFDMA uplink scheduling in ieee 802.11 ax networks,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, pp. 1–6, 2018.

[21] M. Wu, J. Wang, Y.-H. Zhu, and J. Hong, “High throughput resource unit assignment scheme for OFDMA-based WLAN,” in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, pp. 1–8, 2019.

[22] D. W. K. Ng, E. S. Lo, and R. Schober, “Dynamic resource allocation in MIMO-OFDMA systems with full-duplex and hybrid relaying,” *IEEE Trans. Commun.*, vol. 60, no. 5, pp. 1291–1304, 2012.

[23] X. Xiao, X. Tao, and J. Lu, “Energy-efficient resource allocation in LTE-based MIMO-OFDMA systems with user rate constraints,” *IEEE Trans. Veh. Technol.*, vol. 64, no. 1, pp. 185–197, 2015.

[24] W. W. Ho and Y.-C. Liang, “Optimal resource allocation for multiuser MIMO-OFDM systems with user rate constraints,” *IEEE Trans. Veh. Technol.*, vol. 58, no. 3, pp. 1190–1203, 2009.

[25] M. Moretti, L. Sanguinetti, and X. Wang, “Resource allocation for power minimization in the downlink of THP-based spatial multiplexing MIMO-OFDMA systems,” *IEEE Trans. Veh. Technol.*, vol. 64, no. 1, pp. 405–411, 2015.

[26] G. Femenias and F. Riera-Palou, “Scheduling and resource allocation in downlink multiuser MIMO-OFDMA systems,” *IEEE Trans. Commun.*, vol. 64, no. 5, pp. 2019–2034, 2016.

[27] F. Riera-Palou and G. Femenias, “Cluster-based cooperative MIMO-OFDMA cellular networks: Scheduling and resource allocation,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1202–1216, 2018.

[28] S. Kambou, C. Perrine, M. Afif, Y. Pousset, and C. Olivier, “Resource allocation based on cross-layer QoS-guaranteed scheduling for multi-service multi-user MIMO-OFDMA systems,” *Wireless Netw.*, vol. 23, no. 3, pp. 859–880, 2017.

[29] IEEE P802.11ax, “IEEE draft standard for information technology – telecommunications and information exchange between systems local and metropolitan area networks – specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment enhancements for high efficiency WLAN,” *IEEE P802.11ax/D4.0*, pp. 1–746, March 2019.

[30] IEEE 802.11n, “IEEE standard for information technology– local and metropolitan area networks– specific requirements– part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 5: Enhancements for higher throughput,” *IEEE Std. 802.11n*, pp. 1–565, Oct 2009.

[31] IEEE 802.11ac, “IEEE standard for information technology local and metropolitan area networks part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 5: Enhancements for higher throughput,” *IEEE Std. 802.11ac*, 2014.

[32] J. H. Suh, J. Zhu, and O. Aboul-Magd, “System and method for quantization of angles for beamforming feedback,” Feb. 6 2018. US Patent 9,887,749.

[33] A. Perez, *Wi-Fi Integration to the 4G Mobile Network*. John Wiley & Sons Inc., 2018.

[34] Z. Liu, L. Zhang, and Z. Ding, “Exploiting bi-directional channel reciprocity in deep learning for low rate massive MIMO CSI feedback,” *IEEE Wireless Commun. Lett.*, vol. 8, no. 3, pp. 889–892, 2019.

[35] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.

[36] D. Kim, “Normalization methods for input and output vectors in backpropagation neural networks,” *Int. J. Comput. Math.*, vol. 71, no. 2, pp. 161–171, 1999.

[37] Y. Le Cun, I. Kanter, and S. A. Solla, “Eigenvalues of covariance matrices: Application to neural-network learning,” *Phys. Rev. Lett.*, vol. 66, no. 18, p. 2396, 1991.

[38] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” *Int. Conf. Learn. Represent.*, 12 2015.

[39] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proc. Int. Conf. Artif. Intell. Stat.*, pp. 249–256, 2010.

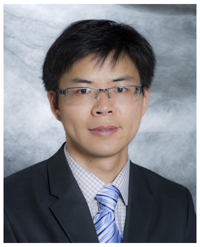
[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[41] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, eds.), pp. 8024–8035, Curran Associates, Inc., 2019.

[42] H. D. Sherali and W. P. Adams, “Reformulation-linearization techniques for discrete optimization problems,” in *Handbook of combinatorial optimization*, pp. 479–532, Springer, 1998.

[43] M. ApS, *The MOSEK optimization toolbox for MATLAB manual. Version 9.0.*, 2019.

Pedram Kheirkhah Sangdeh received his B.Sc. degree in Electrical Engineering from Iran University of Science and Technology, Tehran, Iran, in 2011, and his M.Sc. degree in Electrical Engineering from the College of Engineering, University of Tehran, Tehran, Iran, in 2014. He is currently working toward his Ph.D. degree in the Department of Computer Science and Engineering at Michigan State University, East Lansing, MI, USA. His research interests include design, performance analysis, and implementation of innovative protocols for intelligent wireless networking.



Huacheng Zeng (SM’20) received a Ph.D. degree in Computer Engineering from Virginia Polytechnic Institute and State University (Virginia Tech), Blacksburg, VA. He is currently an Assistant Professor in the Department of Computer Science and Engineering at Michigan State University (MSU), East Lansing, MI. Prior to joining MSU, he was an Assistant Professor of Electrical and Computer Engineering at the University of Louisville, Louisville, KY, and a Senior System Engineer at Marvell Semiconductor, Santa Clara, CA. He is a recipient of the NSF CAREER Award. His research interest is broadly in wireless networking and sensing systems.