Received 10 April 2023; revised 15 May 2023; accepted 4 June 2023. Date of publication 16 June 2023; date of current version 10 July 2023.

Digital Object Identifier 10.1109/OJCOMS.2023.3285790

A Memory-Efficient Learning Framework for Symbol Level Precoding With Quantized NN Weights

ABDULLAHI MOHAMMAD (Student Member, IEEE), CHRISTOS MASOUROS (Senior Member, IEEE), AND YIANNIS ANDREOPOULOS (Senior Member, IEEE)

Department of Electronic and Electrical Engineering, University College London, WC1E 7JEK London, U.K.

CORRESPONDING AUTHOR: A. MOHAMMAD (e-mail: abdullahi.mohammad.16@ucl.ac.uk)

This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/S028455/1 and Grant EP/R035342/1, and in part by the Petroleum Technology Development Fund (PTDF) Overseas Scholarship Scheme, Nigeria, under Award PTDF/ED/PHD/MA/1003/16.

This article was presented in part at the IEEE Global Communications Conference (GLOBECOM), Madrid, Spain, 2021, pp. 1–6 [DOI: 10.1109/GLOBECOM46510.2021.9685567].

ABSTRACT This paper proposes a memory-efficient deep neural network (DNN) framework-based symbol level precoding (SLP). We focus on a DNN with realistic finite precision weights and adopt an unsupervised deep learning (DL) based SLP model (SLP-DNet). We apply a stochastic quantization (SQ) technique to obtain its corresponding quantized version called SLP-SQDNet. The proposed scheme offers a scalable performance vs memory trade-off, by quantizing a scalable percentage of the DNN weights, and we explore binary and ternary quantizations. Our results show that while SLP-DNet provides near-optimal performance, its quantized versions through SQ yield $\sim 3.46\times$ and $\sim 2.64\times$ model compression for binary-based and ternary-based SLP-SQDNets, respectively. We also find that our proposals offer $\sim 20\times$ and $\sim 10\times$ computational complexity reductions compared to SLP optimization-based and SLP-DNet, respectively.

INDEX TERMS Symbol-level-precoding, constructive interference, power minimization, deep neural networks (DNNs), stochastic quantization (SQ).

I. INTRODUCTION

PRECODING using the known channel state information (CSI) at the transmitter has been proven to be an efficient interference management technique in a downlink multiuser multiple-input-single-output (MU-MISO) communication system [1], [2]. The precoding also enables many complex signal processing at the base station (BS), which simplifies users' terminals. Classical block-level precoding (BLP) schemes, where the precoding coefficients are applied across a block of symbols (codewords), have proven to be less computationally expensive than the optimal dirty paper coding (DPC) but suffer performance degradation [3], [4]. Masouros and Alsusa [5] first proposed a method for classifying instantaneous interference into constructive and destructive. The suboptimal precoding strategies that exploit constructive interference (CI) were first introduced [6]. Precoding methods based on optimization are appealing because of their amenability to achieve various performance targets. An optimization-based CI precoding was first introduced using a quadratic optimization strategy in light of vector perturbation precoding [7].

To further improve the performance, a precoding design termed symbol-level-precoding (SLP) that exploits the multiuser interference via CI with the known CSI and transforms it into useful power at the mobile user end has received a lot of attention [8], [9], [10], [11], [12]. The CI-based solution is suitable for practical implementation and has proven that massive multiple-input-multiple-output (m-MIMO) systems can take advantage of the CI with SLP [13], [14], [15]. The idea of CI combined with optimization has been applied in many wireless physical layer designs due to its performance gains over BLP schemes to achieve different objectives, such as transmit power minimization and SINR balancing problems [16], [17], [18], [19], [20], [21]. A closed-form precoding design with optimal performance for a CI exploitation in the MISO downlink for optimization with both strict and relaxed phase rotations was proposed in [20], [22]. While CI-based precoding methods offer superior

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

performance, computing them online on a symbol-by-symbol basis can be computationally demanding.

As a result of the proliferation of machine learning algorithms, the model-driven deep learning (DL) technique that exploits the expert's knowledge has been applied in many wireless communication problems due to its explicability, reliability, and low computational complexity [23], [24], [25]. Therefore, DL-based precoding designs that use domain knowledge have been recently proposed for MU-MISO downlink transmission [26], [27], [28]. However, the drawback of such methods is that the optimization constraints are not directly integrated with the loss function. Furthermore, their performance is bounded by the assumptions and accuracy of the optimal solutions obtained from the optimization algorithm. An unsupervised deep unfolding precoding design termed "SLP-DNet" [29] that utilizes the specifics of the optimization objectives of the precoding problem has been proposed to address these issues and will be used as our benchmark in this work.

Typically, a DL model contains thousands or even millions of learnable parameters, usually stored in a 32-bit floating-point (FP32) numerical presentation, making the model computationally and memory demanding during inference and deployment. To facilitate the online training and deployment of a trained DL model at the device edge, light-weight deep neural network (DNN) designs with lower-precision numerical formats have gained significant attention within the deep learning community, typically applied to image processing applications [30], [31], [32], [33]. However, this concept has not been fully explored in wireless communications. Moreover, the learning technique for traditional precoding and the SLP designs reported in the literature assume ideal neural network implementations and have unquantized weights and activations. Accordingly, those implementations would not apply to the practical scenarios considered in this work with few-bit representations and memory constraints. In this work, we propose a DL model's structural simplification method through weights quantization for SLP design. We present a first attempt to address the potential application of such learning-based SLP approaches and consider a low-bit neural network design approach to the SLP for the power minimization problem. We adopt the DL-based SLP model (SLP-DNet) introduced in [29]. Our contributions are summarized below:

- We propose a memory and complexity efficient DNN approach, applied to the learning-based precoding framework (SLP-DNet) [29]. Specifically, we propose an efficient model simplification via weights compression to accelerate both training and inference and facilitate deployment on the device edge.
- We devise a scalable trade-off between performance and inference complexity, by allowing a percentage of the DNN weights to be quantized, while retaining important weights in full-precision. By tuning the percentage of quantised weights, a scalable trade-off between performance and complexity / memory efficiency is achieved.
- We further introduce a stochastic quantization (SQ) technique that uses the quantization error to alleviate the loss in performance caused by the nonhomogenous quantization errors of the conventional extreme quantization (binary and ternary). In the SQ technique, a fraction of the neural network (NN) weight matrix is quantized to lower resolution while the remaining is retained in its full-precision, resulting in a hybrid quantized weight matrix. The technique yields a memory-efficient DL-based SLP model with a good balance between the performance and the computational complexity.

It should be emphasized the SLP methods are suitable in high interference situations, where the communication standards use lower modulation schemes, such as binary phase-shift keying (BPSK) and quadrature phase-shift keying (QPSK), to guarantee reliability [34]. However, QPSK is used in our experimental simulation due to its robustness to noise, compatibility with higher-order modulations, and practical applicability compared to BPSK. Additionally, constant envelope modulation, such as phase-shift keying (PSK), has become increasingly important due to the recent emergence of large-scale MIMO systems [35]. For example, it is possible to use QPSK to transmit eight bits of information per symbol using 8PSK.

The remainder of the paper is structured as follows: System model and the review of the relevant precoding techniques are presented in Section II. We introduce a technique of designing compressed unsupervised learning-based SLP schemes in Section IV. Simulations and results are presented in Section V. Finally, Section VI concludes the paper.

Notations: We use bold uppercase symbols for matrices, bold lowercase symbols for vectors and lowercase symbols for scalars. Operators $\|\cdot\|_2$, $\|\cdot\|_1$ and $|\cdot|$ denote ℓ_2-norm, ℓ_1-norm and absolute values, respectively. The symbol $\mathbf{\Omega}_i$ represents the i-th trainable parameter associated with DNN layers. Re$\{\cdot\}$ and Im$\{\cdot\}$ represent real and imaginary parts of complex vector/matrix, respectively. Finally, notations $\mathcal{L}(\cdot)$ and $\mathcal{D}(\cdot)$ are used for the loss and parameter update functions, respectively.

II. SYSTEM MODEL AND SYMBOL LEVEL PRECODING

A. SYSTEM MODEL

Consider an MU-MISO downlink transmission in a single cell scenario where an M-antenna base BS serves K single-antenna users. The data is transmitted to the users over flat-fading Rayleigh channel denoted by $\mathbf{h}_i \in \mathbb{C}^{M\times 1}$, and the received signal at the i-th user is expressed as

$$y_i = \mathbf{h}_i^H \sum_{k=1}^{K} \mathbf{u}_k e^{j(\phi_k-\phi_i)} s_i + n_i, \tag{1}$$

where $\mathbf{h}_i$, $\mathbf{u}_k$, n_i represent the channel vector, precoding vector and additive white Gaussian noise of the i-th user.

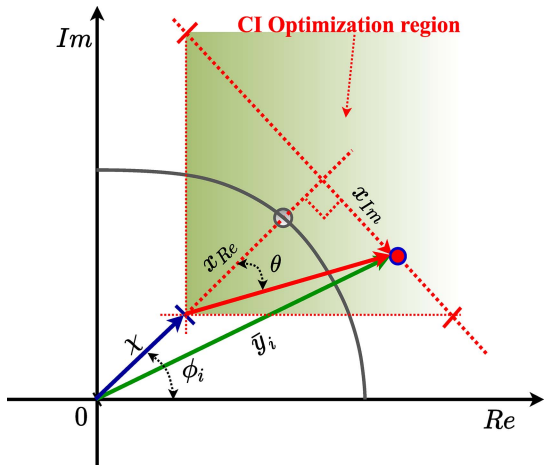


FIGURE 1. Generic geometrical optimization regions for interference exploitation [9].

Therefore, based on the received signal, it can be shown that an SLP power minimization problem is expressed as [9]

$$\min_{\{\mathbf{u_i}\}} \left\| \sum_{k=1}^{K} \mathbf{u}_k e^{j(\phi_k-\phi_1)} \right\|_2^2$$
$$\text{s.t.} \left| \text{Im}\left(\mathbf{h}_i^H \sum_{k=1}^{K} \mathbf{u}_k e^{j(\phi_k-\phi_i)} \right) \right| \leq \left(\text{Re}\left(\mathbf{h}_i^H \sum_{k=1}^{K} \mathbf{w}_k e^{j(\phi_k-\phi_i)} \right) - \sqrt{\Gamma_i \nu_0} \right) \tan\theta, \ \forall i. \quad (2)$$

where $x_{Im} = \text{Im}(\mathbf{h}_i^H \sum_{k=1}^{K} \mathbf{u}_k e^{j(\phi_k-\phi_i)})$ and $x_{Re} = \text{Re}(\mathbf{h}_i^H \sum_{k=1}^{K} \mathbf{u}_k e^{j(\phi_k-\phi_i)})$.

B. SYMBOL LEVEL PRECODING POWER MINIMIZATION

The CI precoding scheme enhances the symbol detection by pushing the received signals away from the constellation detection boundaries without consuming extra transmission power [9]. As an illustration, Fig. 1 shows a symbolic example representing the constellation point $1 + j$ in the QPSK. The green shaded area depicts the constructive region of the constellation based on the least distance (χ) from the decision boundaries, whose value is determined by the SNR constraints. This allows the interfering signals to align with the symbol of interest constructively through precoding vectors. We can observe that if the maximum angle shift in the CI region is zero, the interfering signals overlap entirely on the signal of interest ($\theta = 0$), then the problem reduces to a strict phase angle optimization. It is important to note that the strict phase formulation is not appealing because it yields an increase in the transmission power compared to the corresponding relaxed version [21]. For simplicity, the following are defined according to [9]; $\tilde{\mathbf{h}}_i = \mathbf{h}_i e^{j(\phi_1-\phi_i)} \in \mathbb{C}^{M\times 1}$, $\mathbf{u} = \sum_{k=1}^{K} \mathbf{u}_k e^{j(\phi_k-\phi_1)} \in \mathbb{C}^{M\times 1}$, $\tilde{\mathbf{h}}_{Ri} = \text{Re}(\tilde{\mathbf{h}}_i)$, $\tilde{\mathbf{h}}_{Ii} = \text{Im}(\tilde{\mathbf{h}}_i)$, $\mathbf{u}_R = \text{Re}(\mathbf{u})$ and $\mathbf{u}_I = \text{Im}(\mathbf{u})$. Similarly, we also let $\mathbf{\Phi}_i = [\tilde{\mathbf{h}}_{Ri};\ \tilde{\mathbf{h}}_{Ii}]$, $\mathbf{u}_1 = [\mathbf{u}_R \quad -\mathbf{u}_I]^T$; where $\mathbf{\Upsilon} = \begin{bmatrix} \mathbf{O}_M & -\mathbf{I}_M \\ \mathbf{I}_M & \mathbf{O}_M \end{bmatrix} \in \mathbb{R}^{2M\times 2M}$. For the details and rational of the above definitions, the reader is referred to [6]. The channel matrix can be written as $\tilde{\mathbf{H}} = [\tilde{\mathbf{h}}_1, \ldots, \tilde{\mathbf{h}}_K]$. Therefore, for an $\mathbb{M}$-phase shift keying ($\mathbb{M}$-PSK) modulation scheme, where $\mathbb{M}$ is the modulation index, the optimization-based SLP for a nonrobust multicast power minimization is given by [9]

$$\min_{\{\mathbf{u_1}\}} \ \|\mathbf{u}_1\|_2^2$$
$$\text{s.t.} \ \ \bar{a} \leq \mathbf{\Phi}_i^T \mathbf{\Upsilon} \mathbf{u}_1 \leq \bar{b} \ , \ \forall i. \quad (3)$$

where $\bar{a} = -(\mathbf{\Phi}_i^T \mathbf{\Upsilon} \mathbf{u}_1 - \sqrt{\Gamma_i n_0})\tan\theta$ and $\bar{b} = (\mathbf{\Phi}_i^T \mathbf{\Upsilon} \mathbf{u}_1 - \sqrt{\Gamma_i n_0})\tan\theta$, Γ_i is the target SINR, $\theta = \pm\frac{\pi}{\mathbb{M}}$ is the maximum phase shift in the CI region.

To avoid repetition, we refer the reader to [9], [29] for details and the description of equivalent robust formulations under channel uncertainty.

C. LEARNING-BASED SLP FOR POWER MINIMIZATION (SLP-DNET)

This work is based on the unsupervised deep unfolding framework that unfolds the interior point method (IPM) *'log'* barrier function [36] based on the problem (3) by reformulating it as unconstrained subproblems per user. For simplicity, we drop the i-th subscript, and the unconstrained problem is

$$\min_{\mathbf{u_1}\in\mathbb{R}^{\mathbf{2M}\times\mathbf{1}}} f(\mathbf{u}_1) + \upsilon B(\mathbf{u}_1), \quad (4)$$

where $f(\mathbf{u}_1)$ is the objective function of (3), $B(\mathbf{u}_1) \triangleq -\sum_{i=1}^{t} \ln(g_i(\mathbf{u}_1))$ is the logarithmic barrier function, t is the number of inequality constraints and υ is the Lagrangian multiplier related to the inequality constraints. Since (3) has only one inequality constraint, the function, $g_i(\mathbf{u}_1)$ is defined as $g(\mathbf{u}_1) = (\mathbf{\Phi}_i^T \mathbf{u}_1 - \sqrt{\Gamma_i n_0})\tan\theta - |\mathbf{\Phi}_i^T \mathbf{\Upsilon} \mathbf{u}_1|$.

To derive the learning architecture based on an IPM, we define a proximity barrier of (4) as

$$\text{prox}_{\gamma\upsilon B}(\mathbf{u}_0) = \underset{\mathbf{u_1}\in\mathbb{R}^{\mathbf{2M}\times\mathbf{1}}}{\text{argmin}} \ \frac{1}{2}\|\mathbf{u}_0 - \mathbf{u}_1\|_2^2 + \gamma\upsilon B(\mathbf{u}_1), \quad (5)$$

$\mathbf{u}_0$ is the initial precoding vector and $\gamma > 0$ is the training step size. The precoding vector for every l-th iteration is obtained from the following learning update rule

$$\mathbf{u}_1^{[l+1]} = \text{prox}_{\gamma^{[l]}\upsilon^{[l]}B}\left(\mathbf{u}_1^{[l]} - \gamma^{[l]} \Delta H(\mathbf{u}_1^{[l]}, \lambda^{[l]})\right), \quad (6)$$

where $H(\mathbf{u}_1^{[l]}, \lambda^{[l]}) = \|\mathbf{u}_1\|_2^2 + \lambda\mathbf{u}_1$, and $\Delta H(\mathbf{u}_1^{[l]}, \lambda^{[l]}) = \frac{\partial H(\mathbf{u}_1^{[l]}, \lambda^{[l]})}{\partial \mathbf{u}_1^{[l]}}$. The parameter, λ is introduced as an additional constraint to provide more stability to the learning architecture. Intuitively, NN cascade layers can be formed from (6) as follows

$$\mathbf{u}_1^{[l+1]} = \text{prox}_{\gamma^{[l]}\upsilon^{[l]}B}\left[\left(\mathbf{I}_{2M} - 2\gamma^{[l]}\right)\mathbf{u}_1^{[r]} - \gamma^{[l]}\lambda^{[l]}1^T\right], \quad (7)$$

where $1 \in \mathbb{R}^{1\times 2M}$ is a vector of ones. By letting $\mathbf{W}_l = \mathbf{I}_{2M} - 2\gamma^{[l]}$, $\mathbf{b}_l = -\gamma^{[l]}\lambda^{[l]}1^T$ and $\mathbf{\Pi}_l = \text{prox}_{\gamma^{[l]}\upsilon^{[l]}B}$, the l-layer

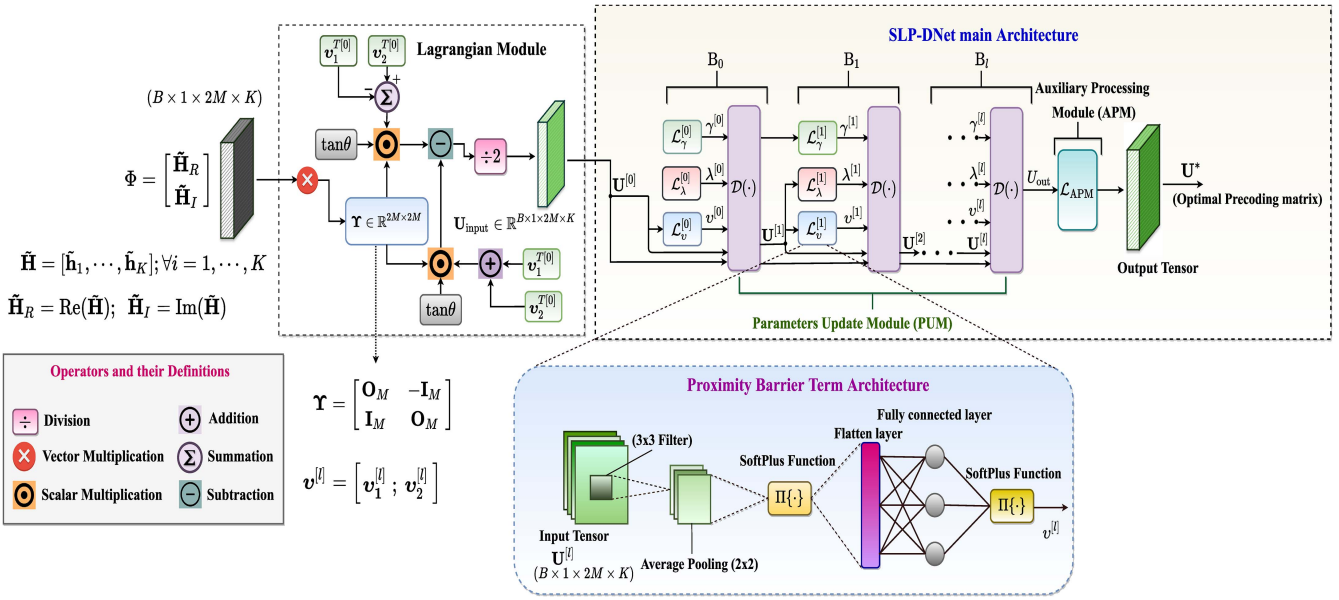


FIGURE 2. Learning-based symbol level precoding (SLP-DNet) Architecture [29].

network $\mathcal{L}^{[l-1]} \cdots \mathcal{L}^{[0]}$ will correspond to the following

$$\mathbf{\Pi}_0(\mathbf{W}_0 + \mathbf{b}_0), \ldots, \mathbf{\Pi}_l(\mathbf{W}_l + \mathbf{b}_l) \tag{8}$$

where $\mathbf{W}_l$ and $\mathbf{b}_l$ are described as weight and bias parameters respectively. The nonlinear activation functions are defined by $\mathbf{\Pi}_l$. The SLP-DNet structure as shown in Fig. 2 is built based on (7) and the Algorithms 1 and 2 described in [29].

As shown in Fig. 2, SLP-DNet has two main units; the parameter update module (PUM) and the auxiliary processing module (APM). The PUM has three core components associated with Lagrangian multiplier (υ), the auxiliary parameter (λ), and the training step-size (γ), which are updated based on (6) are also defined by $\mathcal{D}(\mathbf{u}_1, \boldsymbol{\upsilon}, \gamma, \lambda)$. The structure that is related to the inequality constraint in (3) is the proximity barrier term. It is constructed with one convolutional layer, an average pooling layer, a fully connected layer, and a Soft-Plus layer to constrain the output to a positive real value to satisfy the inequality constraint. The unfolded learning architecture comprises three main parameters that form each learning block for computing υ, λ and γ. These parameters are first initialized randomly such that $\upsilon > 0$, $\lambda > 0$ and $\gamma > 0$, and updated iteratively in each block simultaneously using gradient descent (GD) according to the update rule in [29, Algorithm 1].

The loss function over N batch training samples (batch size or the number of channel realization) is Lagrangian function expressed as

$$\begin{aligned}\mathcal{L}(\mathbf{u}_1, \boldsymbol{\upsilon}_1, \ \boldsymbol{\upsilon}_2) &= \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{u}_1\|_2^2 \\ &+ \frac{1}{N}\sum_{i=1}^{N}\Big(\upsilon_1\Big(\mathbf{\Phi}_i^T\mathbf{\Upsilon}\mathbf{u}_1 - \mathbf{\Phi}_i^T\mathbf{u}_1\tan\theta + \sqrt{\Gamma_i n_0}\Big)\Big) \\ &- \frac{1}{N}\sum_{i=1}^{N}\Big(\upsilon_2\Big(\mathbf{\Phi}_i^T\mathbf{\Upsilon}\mathbf{u}_1 + \mathbf{\Phi}_i^T\mathbf{u}_1\tan\theta - \sqrt{\Gamma_i n_0}\Big)\Big) \\ &+ \frac{\mu}{NL}\sum_{i=1}^{N}\sum_{l=1}^{L}\|\mathbf{\Omega}_l\|_2^2,\end{aligned} \tag{9}$$

where $\mathbf{\Omega}_i$ are the trainable parameters of the l-th layers associated with the weights and biases, and $\mu > 0$ is the penalty parameter that controls the bias and variance of the trainable coefficients. The optimal precoder is obtained from the Lagrangian function (9) as

$$\mathbf{u}_1 = \frac{\left(\boldsymbol{\upsilon}_1^T + \boldsymbol{\upsilon}_2^T\right)\cdot \mathbf{\Phi}_i\tan\theta - \left(\boldsymbol{\upsilon}_1^T - \boldsymbol{\upsilon}_2^T\right)\cdot \mathbf{\Upsilon}^T\mathbf{\Phi}_i\tan\theta}{2}. \tag{10}$$

D. ROBUST SLP-DNET

The exact CSI is usually not known in practice. Therefore, an ellipsoid ξ is often considered to model the user's actual channel in the uncertainty region such that the channel error is within it [37]. By considering the ellipsoid ξ, the actual channel can be expressed as $\hat{\mathbf{h}}_i = \tilde{\mathbf{h}}_i + \tilde{\boldsymbol{e}}_i \ \forall k$, where $\tilde{\mathbf{h}}_i$ is the known CSI at the BS and $\tilde{\boldsymbol{e}}_i$ is the channel error. The model of the uncertainty ellipsoid with the center $\hat{\mathbf{h}}_i$ is thus [9]

$$\xi = \left\{\tilde{\mathbf{h}}_i + \tilde{\mathbf{e}}_i|_{\|\tilde{\mathbf{e}}_i\| \le 1\|}\right\}, \tag{11}$$

where the channel error is given by $\{\tilde{\mathbf{e}}_i : \|\tilde{\mathbf{e}}_i\|_2^2 \le \varsigma_i^2\}$ bounded by ς_i^2. Based on this, the multi-cast CI power minimization problem for the worst-case CSI error is given by

$$\begin{aligned}\min_{\{\mathbf{u}\}} \ & \|\mathbf{u}\|_2^2 \\ \text{s.t.} \ & \left|\text{Im}\left(\hat{\mathbf{h}}_i^T\mathbf{u}\right)\right| - \left(\text{Re}\left(\hat{\mathbf{h}}_i^T\mathbf{u}\right) - \sqrt{\Gamma_i n_0}\right)\tan\theta \le 0, \\ & \forall\|\tilde{\mathbf{e}}_i\|_2^2 \le \varsigma_i^2, \ \forall i.\end{aligned} \tag{12}$$

To guarantee the constraint in (12) is satisfied, it is modified to yield the robust SLP as follows [9]

$$\begin{aligned} \min_{\{\mathbf{u_2}\}} \ & \|\mathbf{u}_2\|_2^2 \\ \text{s.t.} \ & \boldsymbol{\Phi}^T\mathbf{Q}_1\mathbf{u}_2 + \varsigma\|\mathbf{Q}_1\mathbf{u}_2\|_2 + \sqrt{\Gamma n_0}\tan\theta \le 0 \ \forall i \\ & \boldsymbol{\Phi}^T\mathbf{Q}_2\mathbf{u}_2 + \varsigma\|\mathbf{Q}_2\mathbf{u}_2\|_2 + \sqrt{\Gamma n_0}\tan\theta \le 0 \ \forall i. \end{aligned} \tag{13}$$

In a similar fashion to the nonrobust SLP-DNet, we can derive a CSI-robust SLP-DNet from the robust SLP formulation under worst-case CSI-error. For convenience, we introduce new notations as follows: $\mathbf{Q}_1 = (\boldsymbol{\Upsilon} - \mathbf{I}_{2M}\tan\theta)$ and $\mathbf{Q}_2 = (\boldsymbol{\Upsilon} + \mathbf{I}_{2M}\tan\theta)$ and ς^2 is the CSI error bound. (13) is a second order cone programming (SOCP) and can be solved using convex optimization software package.

It is important to note that the structure of the robust SLP-DNet is obtained by following similar steps from (4)-(6) of Section II-C by transforming (13) to its equivalent unfolded IPM *'log'* barrier form. The loss function is obtained from the Lagrangian of (13) as

$$\begin{aligned} \mathcal{L}_{\text{robust}}(\mathbf{u}_2, \ \boldsymbol{\upsilon}_1, \ \boldsymbol{\upsilon}_2) = & \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{u}_2\|_2^2 + \\ & \frac{\boldsymbol{\upsilon}_1}{N}\sum_{i=1}^{N}\left(\varsigma^2\|\mathbf{Q}_1\mathbf{u}_2\|_2^2 - \left(\sqrt{\Gamma n_0}\tan\theta - \boldsymbol{\Phi}^T\mathbf{Q}_1\mathbf{u}_2\right)^2\right) \\ & + \frac{\boldsymbol{\upsilon}_2}{N}\sum_{i=1}^{N}\left(\varsigma^2\|\mathbf{Q}_2\mathbf{u}_2\|_2^2 - \left(\sqrt{\Gamma n_0}\tan\theta - \boldsymbol{\Phi}^T\mathbf{Q}_2\mathbf{u}_2\right)^2\right) \\ & + \frac{\mu}{NL}\sum_{i=1}^{N}\sum_{i=1}^{L}\|\boldsymbol{\Omega}_i\|_2^2. \end{aligned} \tag{14}$$

where $\left[\|\mathbf{Q}_1\|_2^2 \ \|\mathbf{Q}_2\|_2^2\right] = \tilde{\mathbf{q}}_{\text{norm}}$, $\left[\mathbf{Q}_1 \ \mathbf{Q}_2\right] = \tilde{\mathbf{Q}}$ and $\left[\boldsymbol{\upsilon}_1 \ \boldsymbol{\upsilon}_2\right] = \tilde{\boldsymbol{\upsilon}}$.

The optimal precoder can be easily obtained from (14)

$$\mathbf{u}_2 = -\boldsymbol{\Phi}\tilde{\mathbf{Q}}\tilde{\boldsymbol{\upsilon}}^T\mathbf{X}^{-1}\sqrt{\Gamma n_0}\tan\theta, \tag{15}$$

where $\mathbf{X} = (\mathbf{I}_{2M} + \tilde{\mathbf{q}}_{\text{norm}}\tilde{\boldsymbol{\upsilon}}^T(\varsigma^2 - \boldsymbol{\Phi}^T\boldsymbol{\Phi}))$. Note that the Lagrange multipliers $\boldsymbol{\upsilon}_1$ and $\boldsymbol{\upsilon}_2$ are associated with the barrier term and are randomly initialized from a uniform distribution.

III. PRELIMINARIES OF NN WEIGHT QUANTIZATION

Traditionally, DNN is designed with full-precision weights and activations. This can result in significant memory consumption and computational complexity. For this reason, there has been a recent drive to reduce the DNN model size [30]. DNN acceleration techniques can be broadly classified into three categories: structured simplification [38], optimized implementation [39] and quantization [30]. Among them, quantization is most appealing because, most multiply-accumulate (MAC) operations required to compute the neurons' weighted sums are replaced by simple binary operations (bit-wise or XNOR operations). Quantization improves both training and inference efficiencies; and reduces hardware requirements during model deployment on the edged-devices.

Typically, the weights of l-th layer DNN architecture are represented by $\mathcal{W}_l = \{\mathbf{W}_i, \ldots, \mathbf{W}_m\}$ for $\forall \ i = 1, \ldots, m$, where m is the number of kernels/filters (output channels) [31].

1) BINARY WEIGHTS

The full-precision weights are converted to ($\mathbf{B}_w \in \{+1, -1\}^n$). A full-precision 32-bit weight matrix is binarized as follows [40]

$$\mathbf{B}_{\text{w}} = sign(\mathbf{W}) = \begin{cases} +1 & \text{if } \mathbf{W} \ge 0 \\ -1 & \text{otherwise,} \end{cases} \tag{16}$$

A more robust binarized weight "BWN" is proposed as an extension of a straightforward binary network (Binary Connect) by introducing a real scaling factor $\beta \in \mathbb{R}^+$ such that $\mathbf{W} \approx \beta\mathbf{B}_{\text{w}}$ by solving an optimization problem [31]

$$J(\mathbf{B}_{\text{w}}, \beta) = \underset{(\beta)}{\text{argmin}} \ \|\mathbf{W} - \beta\mathbf{B}_{\text{w}}\|_2^2, \tag{17}$$

and this yields

$$\beta^* = \frac{1}{n}\|\mathbf{W}\|_1. \tag{18}$$

2) TERNARY WEIGHTS

A ternary weighted network (TWN) is the one in which an extra 0 state is introduced into BWN to solve the following optimization problem [41]

$$\begin{cases} \beta^*, \mathbf{B}_{\mathbf{W}}^* = \underset{\beta, \ \mathbf{B}_{\text{w}}}{\text{argmin}} \ J(\beta, \ \mathbf{B}_{\text{w}} \ ) = \|\mathbf{W} - \beta\mathbf{B}_{\text{w}}\|_2^2 \\ \text{s.t.} \qquad \beta \ge 0, \ \mathbf{B}_{\text{w}} \in \{-1, \ 0, \ +1\}^n, \end{cases} \tag{19}$$

and solving (19) gives

$$\mathbf{B}_{\text{w}}^* = \begin{cases} +1, & \text{if } \mathbf{W} > \delta \\ 0, & \text{if } |\mathbf{W}| \le \delta \\ -1, & \text{if } \mathbf{W} < -\delta, \end{cases} \tag{20}$$

where $\delta = \frac{0.7}{n}\sum_{i=1}^{n}|\mathbf{W}|$ and $\beta^* = \frac{1}{\mathbf{I}_\delta}\sum_{i\in\mathbf{I}_\delta}|\mathbf{W}|$, $\mathbf{I}_\delta = \{|\mathbf{W}| > \delta\}$ is the cardinality of set $\mathbf{I}_\delta$.

IV. PROPOSED LOW-BIT SLP-DNET DESIGN

A. LOW-BIT WEIGHTS AND STOCHASTIC DIVISION

The existing works on low-bit DNNs design focus only on reducing the bit-widths of the weights and activations to speed up the training and inference times and also improve memory efficiency. However, in low-bit DNNs designs, the impact of quantization on the performance of the learning algorithm has not been fully explored and understood. In this work, we adopt a quantization technique proposed in [33] and propose a simple linear probability function of selecting the filter weights to be quantized for designing a low-bit scalable learning-based precoder. The weight matrix of each layer of the DNN can be expressed as: $\mathcal{W} = \{\mathbf{W}_1, \ldots, \mathbf{W}_n\}$.

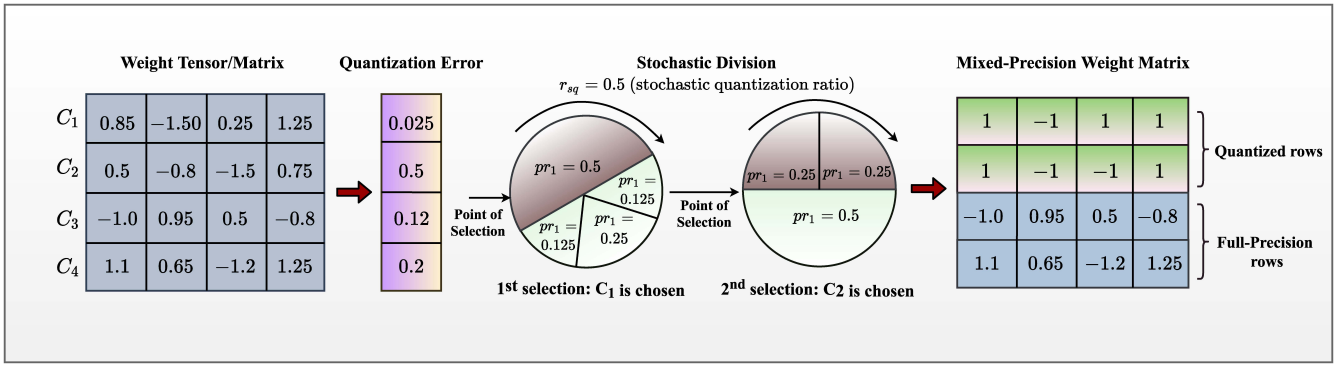


FIGURE 3. Stochastic quantization weight matrix partitioning procedure.

Here, the rows of the weight matrix are partitioned into two parts according to the following

$$\mathcal{W} = \{\mathcal{W}_q,\ \mathcal{W}_f\}, \tag{21}$$

where $\mathcal{W}_q = \{\mathbf{W}_{q1}, \ldots, \mathbf{W}_{qM}\}$ and $\mathcal{W}_f = \{\mathbf{W}_{f1}, \ldots, \mathbf{W}_{fN}\}$ represent the quantized and unquantized parts of the weight respectively, and should satisfy the condition below

$$\mathcal{W} = \mathcal{W}_q \cup \mathcal{W}_f \text{ and } \mathcal{W}_q \cap \mathcal{W}_f = \emptyset. \tag{22}$$

As seen from (21), one subset of the weight $\mathcal{W}_q$ is quantized to a low bit while the remaining $\mathcal{W}_f$ is kept in its full-precision form, so that the entire weights matrix is composed of both binary and floating-point values. It is important to note that bias is also quantized as the weight, but it is not related to any input tensor (i.e., it is not used for weight-input computations). Therefore, the selection of the fraction of the filter weight to be quantized does not include the bias because the predominant computation involves a convolution operation between the filter weight and the input tensor. Note that a fully quantized DNN can be obtained by setting $\mathcal{W}_f$ to a null set.

Suppose r_{sq} is the quantization ratio (QR) (i.e., the percentage of weights quantized as a fraction of the total weights in the DNN), and n is the length of the weight matrix (number of elements), the number of elements in the quantization group is $M_q = r_{sq}n$ while that of unquantized part is $M_f = (1 - r_{sq})n$. The QR is gradually increased to 100% until the entire network is finally quantized. We assign a probability $\mathbf{pr} \in \mathbb{R}^n$ to each row of the matrix $\mathcal{W}$, where each value of pr_i in $\mathbf{pr}$ represents the likelihood that the corresponding i-th row will be quantized. To select the channel to be quantized, we adopt a lottery disc algorithm as in [33]. It can be observed in Fig. 3 that each sector of the disc represents a probability of selecting a channel (row of weight matrix). The disc is rotated by choosing a value from the uniform distribution whose magnitude is slightly above the probability value. After every selection, the probability is reset (i.e., $pr_i = 0$) to ensure that a channel is selected without replacement as summarized in Algorithm 1.

Algorithm 1 Circular Lottery Algorithm for Weight Matrix Division

Input: r_{sq} Stochastic Quantization ratio and Weight matrix ($\mathcal{W}$)
Output: $\mathcal{W}_q$ and $\mathcal{W}_f$

Initialization:
$\mathcal{W}_q = \mathcal{W}_r = \emptyset$
Compute QP function $\mathbf{pr} \in \mathbb{R}^n \forall i\ \{i = 1, \cdots, n\}$ based on (23)
$M_q = r_{sq}n$
for $j = 1$ to M_q **do**
  $\hat{\mathbf{p}}r = \frac{\mathbf{pr}}{\|\mathbf{pr}\|_1}$ (normalized probability)
  Select a random value $\vartheta_j \in \{0, 1\}$ from a random uniform distribution
  Set $s_j = 0$ and $i = 0$, s_j gathers the normalized probability $\hat{\mathbf{p}}r$
  while $s_j < \vartheta_j$ **do**
    $i = i + 1$
    $s_j = s_j + \hat{pr}_i$; $\hat{pr}_i$ is the i-th member in $\hat{\mathbf{pr}}_i$
  end while
  Compute: $\mathcal{W}_q = \mathcal{W}_q \cup \{\mathcal{W}\}$; shows $\mathcal{W}_q$ is the subset of $\mathcal{W}$
  Reset $pr_i = 0$ {This is to avoid *i-th* channel weight from being selected again}
  Compute: $\mathcal{W}_r = \mathcal{W} \setminus \mathcal{W}_q$ (complement of W_q)
end for

B. QUANTIZATION ERROR AND QUANTIZATION PROBABILITY

Recall that classical binarized DNNs suffer a significant performance loss due heterogeneous nature of the quantization error (QE) over the entire network. The performance can, however, be improved by stochastically selecting the filter or channel weight matrix to be quantized using a random probability distribution based on the QE between the full-precision and quantized weights as follows

$$e_j = \frac{\left\|\mathbf{W}_j - \mathbf{Q}_j^*\right\|_1}{\|\mathbf{W}_j\|}; \tag{23}$$

where $\mathbf{Q}_j^*$ could be binary or ternary based on (18) or (20).

We define the vector of the *n-th* row weight matrix of a given layer as $\mathbf{e} = [e_1, \ldots, e_n]$. The quantization probability is formulated such that a higher probability is assigned to filter/weights if the quantization error is small because quantizing these weights does not yield a significant loss of accuracy or performance. For a given weight matrix, QR, and quantization probability (QP), a channel is randomly sampled without replacement using a circular lottery Algorithm 1. From this, we can observe that the QP function is inversely proportional to QE and is defined as $f_p = \frac{1}{e+\delta}$, where $\delta = 10^{-6}$ to avoid possible numerical overflow. The QP function is monotonically non-increasing to prioritize the selection of the channels/weights to be quantized. Different monotonically non-increasing functions are:

- Uniform function: $pr_i = \frac{1}{n}$, n is the number of the neurons or length of the rows of each layer weight matrix.
- Linear function: $pr_i = \frac{f_{p_i}}{\sum_i f_{p_j}}$.
- Half-Gaussian function: $pr_i = \frac{\sqrt{2}}{\sigma\sqrt{\pi}} \exp\left(\frac{-f_{p_i}^2}{2\sigma^2}\right)$.
- Softmax function: $pr_i = \frac{\exp(f_{p_i})}{\sum_i \exp(f_{p_i})}$.

The simplest of these QP functions is uniform or constant function but is not appealing because it is independent of the QE and therefore ignores the random quantization proposition. The most intriguing of all is the half-Gaussian function because of the extra parameter (σ), which can be learned but is more complicated. The linear and softmax functions have been found to yield nearly the same performance, but the former is simpler to implement. Accordingly, in this work, we use the linear function because it balances between performance and simplicity.

C. LOW-BIT ACTIVATION FUNCTION

The inputs to convolutional and fully connected layers are often the outputs of the previous layers' activations. In many low-bit DNNs designs, the activation layer is often left in its full-precision. However, quantizing the activation layer is crucial in replacing the floating-point operations with more efficient binarization. The conventional activation functions such as *"Relu"* may not be suitable for low-bit DNNs [42]. Therefore, the activations are quantized from 32-bit($\mathbf{u}_{32}$) to $k-$bit according to the function

$$\mathbf{W}_b = \frac{round\big((\mathbf{W}_{32} - x) \cdot \big(2^k - 1\big)/(y - x)\big)}{\big(2^k - 1\big)} \tag{24}$$

where $\mathbf{W}_{32}$ is the floating-point activation bounded by the input dimension (x, y) and $k = 2$. The activations are not stochastically quantized because, unlike in weights, the activations do not have learning parameters.

D. MODEL TRAINING AND INFERENCE

1) SLP-DNET AND CLASSICALLY QUANTIZED SLP-DNET

The SLP-DNet is trained the same way as its corresponding classically quantized versions based on binary and ternary bits (SLP-DBNet and SLP-DTNet). Each PUM block contains three main components and is trained block-wise for k-th number of iterations. Similarly, APM is trained for r-th iterations, and the number of training iterations of the PUM and APM may not necessarily be equal. The PUM is trained for 20 iterations and the APM for 10 iterations. We use a greedy approach for training the PUM. The first PUM block (B_0) training stops after a fixed number of iterations. The output is saved and used as an input to train the following block (B_1). When the training of B_1 is complete, its output is used to train the next block, etc. The output of the last PUM is the input to APM (see Fig. 2), and is trained as one block for the same or a different number of iterations as PUM. This training strategy is chosen because it requires low memory. We observed that training PUM and APM beyond 20 and 10 iterations did not improve the performance further. We modify the learning rate by a factor $\alpha \in \mathbb{R}^+$ for every training step to improve the training efficiency using a stochastic gradient descent (SGD) algorithm with Adam optimizer [43].

2) STOCHASTIC QUANTIZED SLP-DNET(SLP-DSQNET)

The SLP-DSQNet training is slightly different from that of SLP-DNet. The training is summarized in four stages: stochastic weight matrix division, forward propagation, backward propagation, and parameter update. Given QR, the weight matrix is partitioned into a quantization group and a full-precision group using Algorithm 1. A hybrid weight is then formed containing the quantized and the full-precision weights, and it provides a better gradient direction than pure quantized weights. If $\tilde{\mathcal{W}}_{qf}$ is the composite weight matrix, the weight update with respect to the composite gradients is given by $\mathcal{W}^{r+1} = \mathcal{W}^r - \eta \frac{\partial \mathcal{L}(\cdot)}{\partial \tilde{\mathcal{W}}_{qf}^r}$. We train the network with different QRs, which are fixed for all the training iterations and inference.

The back-propagation through the quantization function results in zero gradients due to the thresholding that summarizes the activations or outputs into binary values. This lack of gradient results in the network not learning anything. However, a straight-through estimator (STE) [30] is used in the backward pass to solve this problem. Intuitively, the STE uses the identity function to approximates quantization in the backprop as if the function was an identity function expressed as [44]

$$clip(x, -1, 1) = \max(-1, \min(1, x)). \tag{25}$$

The learning is performed in an unsupervised fashion in which the loss function is the Lagrangian function's statistical mean over the training batch. During the inference, a feed-forward pass is performed over the whole layers using the learned Lagrangian multipliers to compute the precoding vector using (10) and (15) for nonrobust and robust SLP formulations. Training can be done with an arbitrary SINR value, which may not guarantee good learning. However, we train the model with different SINRs to determine the appropriate training SINR to improve learning ability. Therefore,

the training SINR is drawn from a random uniform distribution to enable learning across a wide range of SINR values.

E. COMPUTATIONAL COMPLEXITY ANALYSIS

This subsection presents the analytical evaluations of the computational costs of the proposed SLP-DSQNet precoding schemes and compares them with SLP-DNet, the conventional BLP, and the classical SLP optimization-based methods. The complexities are computed in terms of the number of real arithmetic operations involved. To derive the analytical complexity of the optimization-based SLP, we first convert the second-order cone programming (SOCP) (3) into standard linear programming (LP) as follows

$$\begin{aligned} &\min_{\{\mathbf{u}_1\}} \ \|\mathbf{u}_1\|_2^2 \\ &\text{s.t.} \ \ |\mathbf{\Phi}_i^T \mathbf{\Upsilon} \mathbf{u}_1| \le \bar{b}, \ \ \forall i. \end{aligned} \tag{26}$$

where $\bar{b} = (\mathbf{\Phi}_i^T \mathbf{\Upsilon} \mathbf{u}_1 - \sqrt{\Gamma_i n_0})\tan\theta$. To convert (26) to its equivalent LP, we introduce new optimization variables

$$\begin{aligned} &\min_{\{\mathbf{x},\mathbf{d}\}} \ \mathbf{d}^T\mathbf{x} \\ &\text{s.t.} \ \ \mathbf{d}_k^T\mathbf{x} \le -\tan\theta\sqrt{\Gamma_i n_0}, \ \ \forall i \end{aligned} \tag{27}$$

where $\mathbf{d} = [0 \ \ \mathbf{u}_1^T]^T \in \mathbb{R}^{(2M+1)\times 1}$, $\mathbf{x} = [1 \ \mathbf{u}_1]^T \in \mathbb{R}^{(2M+1)\times 1}$, $\mathbf{d}_i = [|\mathbf{\Phi}_i^T \mathbf{\Upsilon} \mathbf{u}_1| \quad -\mathbf{\Phi}_i^T \tan\theta]^T \in \mathbb{R}^{(2M+1)\times 1}$.

Given the optimal target accuracy, $\epsilon > 0$, the complexity of solving convex optimization via IPM is characterized by the formation (C_{form}) and factorization (C_{fact}) of the matrix coefficients with $\bar{n}$ linear equations having $\bar{n}$ unknowns and is given by [45]

$$C_{\text{total}} = (C_{\text{form}} + C_{\text{fact}}) \times \ln\left(\frac{1}{\epsilon}\right) \sqrt{\sum_{j=1}^{M_{\text{lc}}} Q_j + 2M_{\text{sc}}} \tag{28}$$

where Q represents the constraint's dimension, M_{lc} and M_{sc} denote the numbers of linear inequality matrix and second order cone (SOC) constraints, respectively. Therefore, the overall complexity is

$$C_{\text{total}} = \underbrace{\left[\underbrace{\bar{n}\sum_{j=1}^{M_{\text{lc}}} Q_j^3}_{\text{due to } M_{\text{lc}}} + \bar{n}^2 \sum_{j=1}^{M_{\text{lc}}} Q_j^2 + \underbrace{\bar{n}\sum_{j=1}^{M_{\text{sc}}} Q_{j=1}^2}_{\text{due to } M_{\text{sc}}} + \bar{n}^3\right]}_{C_{\text{form}}+C_{\text{fact}}} \times \ln\left(\frac{1}{\epsilon}\right) \sqrt{\sum_{j=1}^{M_{\text{lc}}} Q_j + 2M_{\text{sc}}}. \tag{29}$$

It can be observed that (27) has K constraints with dimension $2M+1$. Therefore, using (29), the total computational cost is obtained as

$$C_{\text{total}} = \sqrt{2M+1}[\bar{n}(2M+1) + \bar{n}(2M+1)^2 + \bar{n}^3] \ln\left(\tfrac{1}{\epsilon}\right). \tag{30}$$

By following similar principles and steps above, we can obtain the complexities of the robust SLP and the conventional BLP schemes.

On the other hand, to determine the complexities of our proposed precoders, we first evaluate the complexities of the learning modules (PUM and APM) in terms of arithmetic operations involved. For PUM, there are three convolution blocks. The feature map determines the arithmetic operations for a convolution layer and is given by the number of multiplications and additions involved in the convolution operation. The number of operations in a given convolutional layer is

$$C_{\text{conv}} = \left(C_{\text{in}} k_{\text{f}}^2 + (C_{\text{in}} k_{\text{f}}^2 - 1) + 1\right) C_{\text{out}} N_{\text{w}} N_{\text{h}} \tag{31}$$

where N_{h}, N_{w}, k_{f}, C_{in} and C_{out} denote the height, width of the input layer tensor, filter size, number of input and output channels, respectively. It is important to note that only the first and second convolutions are quantized, while the last convolution is not to avoid losing essential features of the output precoder. Since in our proposed approach, the layer weight matrix contains both floating points and quantized entries, then the quantization approximation of convolution has $\frac{1}{32}(c_{\text{in}} k_{\text{f}}^2 N_{\text{w}} N_{\text{h}} c_{out}) \times QR$ binary operations and $(c_{\text{in}} k_{\text{f}}^2 N_{\text{w}} N_{\text{h}} c_{out}) \times (1 - QR)$ non binary operations based on (31). Using these expressions, we obtain the generic complexity of the PUM as

$$C_{\text{PUM}} = \underbrace{\frac{1}{32}\sum_{l=1}^{L} N_{\text{h}}^{[l-1]} N_{\text{w}}^{[l-1]} \left[C_{\text{in}}^{[l-1]} f^{[l]2}\right] C_{\text{out}}^{[l]} (QR) +}_{\text{binary operations}}$$

$$\underbrace{\sum_{l=1}^{L} N_{\text{h}}^{[l-1]} N_{\text{w}}^{[l-1]} \left[C_{\text{in}}^{[l-1]} f^{[l]2}\right] C_{\text{out}}^{[l]} (1 - QR)}_{\text{floating point operations}}. \tag{32}$$

Similarly, the APM's complexity is determined by the cost of the feed-forward pass of the shallow CNN, as shown in Table 3 and the *'log'* barrier that form the barrier term.

$$\begin{aligned} C_{\text{APM}} = &\sum_{l=1}^{L_{\text{cv}}} N_{\text{h}}^{[l-1]} N_{\text{w}}^{[l-1]} \left[C_{\text{in}}^{[l-1]} f^{[l]2}\right] C_{\text{out}}^{[l]} + \\ &\sum_{j=1}^{L_{\text{fc}}} \left(2N_{\text{in}}^{[j-1]} + 1\right) N_{\text{out}}^{[i]} + \\ &C_{\text{log-barrier}} \end{aligned} \tag{33}$$

where L_{cv} and L_{fc} are the number of convolution and fully connected layers, respectively. Based on the matrix/vector multiplications, the square absolute and l_2 norm values, the number of arithmetic operations involved in computing the terms in the *'log'* barrier functions for SLP-DNet and robust SLP-DNet are obtained as $4M^2K + 2MK + K$ and $8M^2K + 4MK + 6K$, respectively.

Finally, we use the information in Tables 3 and 4 along with (32) and (33) to obtain the complexity of

TABLE 1. Complexity analysis of proposed SLP-DSQNet and benchmark SLP schemes.

Method	Arithmetic Operations (term; $\bar{n} = \mathcal{O}(2KM)$)	Complexity Order ($\bar{n} = M = K$)
Conventional BLP	$\sqrt{(4M+K+2)}\left[\bar{n}(2M+1)+\bar{n}(2M+1)^2+\bar{n}(K+1)^2+\bar{n}^3\right]\ln\left(\frac{1}{\epsilon}\right)$	$\mathcal{O}(\bar{n}^{6.5})$
SLP Optimization-based	$\sqrt{2M+1}\left[\bar{n}(2M+1)+\bar{n}(2M+1)^2+\bar{n}^3\right]\ln\left(\frac{1}{\epsilon}\right)$	$\mathcal{O}(\bar{n}^{6.5})$
SLP-DNet	$2704K^2M+4M^2K+430KM-K$	$\mathcal{O}(\bar{n}^3)$
SLP-DBNet	$127K^2M+4M^2K+7KM-K-\frac{7}{8}$	$\mathcal{O}(\bar{n}^3)$
SLP-DTNet	$271K^2M+4M^2K+\frac{77}{2}KM-K-\frac{7}{8}$	$\mathcal{O}(\bar{n}^3)$
SLP-DSQBNet	$2704K^2M+430KM+4M^2K-K-\left[2577K^2M+423KM+\frac{7}{8}\right]\times QR$	$\mathcal{O}(\bar{n}^3)$
SLP-DSQTNet	$2704K^2M+430KM+4M^2K-K-\left[2433K^2M+\frac{783}{2}KM+\frac{7}{8}\right]\times QR$	$\mathcal{O}(\bar{n}^3)$
Robust Conventional BLP	$\sqrt{2K(2M+1)}\left[\bar{n}K(2M+1)^3+\bar{n}^2K(2M+1)^2+\bar{n}^3\right]\ln\left(\frac{1}{\epsilon}\right)$	$\mathcal{O}(\bar{n}^{7.5})$
Robust SLP Optimization-based	$\sqrt{2(2M+1)}\left[2\bar{n}K(2M+1)^2+\bar{n}^3\right]\ln\left(\frac{1}{\epsilon}\right)$	$\mathcal{O}(\bar{n}^{6.5})$
Robust SLP-DNet	$2704K^2M+8M^2K+432KM+8M^2K+6K-2$	$\mathcal{O}(\bar{n}^3)$
Robust SLP-DBNet	$127K^2MK+8M^2K+9KM+6K-\frac{9}{8}$	$\mathcal{O}(n^3)$
Robust SLP-DTNet	$271K^2M+8M^2K+\frac{81}{2}KM+6K-\frac{9}{8}$	$\mathcal{O}(\bar{n}^3)$
Robust SLP-DSQBNet	$2704K^2M+8M^2K+432KM+6K-2-\left[2577K^2M+423KM+\frac{7}{8}\right]\times QR$	$\mathcal{O}(\bar{n}^3)$
Robust SLP-DSQTNet	$2704K^2M+8M^2K+432KM+6K-2-\left[2433K^2M+\frac{783}{2}KM+\frac{7}{8}\right]\times QR$	$\mathcal{O}(\bar{n}^3)$

TABLE 2. Simulation settings.

Parameters	Values
Number of BS antennas	4
Number of users	4
Training Samples	50000
Batch Size (B)	200
Test Samples	2000
Training SINR range	0.0dB - 45.0dB
Test SINR range (i-th user SINR)	0.0dB - 35.0dB
Optimizer	SGD with Adam
Initial Learning Rate, η	0.001
Learning Rate decay factor, α	0.65
Lower bit Activation	bits-width, $k = 2$
Number of blocks in the PUM	$B_l = 2$
Training Iterations in the PUM per block	20
Training iterations for the APM	10

TABLE 3. Proximity barrier term NN layout.

Layer	Parameter, kernel size $= 3 \times 3$
Input Layer	Input size (B, 1, $2M$, K)
Layer 1: Convolutional	Size (B, 20, $2M$, K^2); zero padding
Layer 2: Average Pooling	Size ((1, 1), stride = (1, 1))
Layer 3: Activation	Soft-Plus
Layer 4: Flat	Size (B $\times$ 40 $\times$ K^2)
Layer : Fully-connected	Size(B $\times$ 40 $\times$ K^2, 1)
Layer 5: Activation	Soft-Plus function

TABLE 4. An APM NN architecture.

Layer	Parameter, kernel size $= 3 \times 3$
Input Layer	Input size (B, 1, $2M$, K)
Layer 1: Convolutional	Size (B, 16, $2M$, K), dilation = 1 and unit padding
Layer 2: Batch Normalization	eps $= 10^{-6}$, momentum $= 0.1$
Layer 3: Activation	PReLu/k-bit function
Layer 4: Convolutional	Size (B, 8, K, $2KM$), dilation = 1 and unit padding
Layer 5: Batch Normalization	eps $= 10^{-6}$, momentum $= 0.1$
Layer 6: Activation	PReLu/k-bit function
Layer 7: Convolutional	Size (B, 1, $2KM$, 1), dilation = 1 and unit padding

SLP-DSQBNet as follows

$$C_{\text{SQB}} = 2704K^2M + 430KM + 4M^2K - K - \left[2577K^2M + 423KM + \frac{7}{8}\right] \times QR. \quad (34)$$

We can obtain SLP-DSQTNet's complexity from (34) by introducing additional '0' state, and this additional bit yields

$$C_{\text{SQT}} = 2704K^2M + 430KM + 4M^2K - K - \left[2433K^2M + \frac{783}{2}KM + \frac{7}{8}\right] \times QR \quad (35)$$

We observe that by substituting $QR = 0$ in (34) or (35), we can obtain the complexity of SLP-DNet. Similarly, the complexities of SLP-DBNet and SLP-DTNet are also found by substituting $QR = 1$ in (34) and (35), respectively. Table 5 shows the complexities of the proposed and benchmarks precoding schemes. For illustration, we use the case of symmetry, where ($M = K = \bar{n}$), and show that our proposals have a considerably lower computational complexity of $\mathcal{O}(\bar{n}^3)$. In contrast, the optimization-based SLP and conventional BLP methods have $\mathcal{O}(\bar{n}^{6.5})$ and $\mathcal{O}(\bar{n}^{7.5})$ computational complexities, respectively. While our proposed schemes have the same order of complexity as SLP-DNet (see Table 1), the number of arithmetic operations involved in their computations is lower than that of the SLP-DNet due to the presence of binary operations.

V. SIMULATION RESULTS AND DISCUSSION

A. SIMULATION SET-UP

We consider a downlink situation in which the BS is equipped with four antennas ($M = 4$) that serve K single users; and assume a single cell. We obtain the dataset from the channel realizations randomly generated from a normal distribution with zero mean and unit variance. The dataset is reshaped and converted to real number domain using the following expression $\mathbf{\Phi} = \left[\tilde{\mathbf{h}}_{Ri};\ \tilde{\mathbf{h}}_{Ii}\right]$ as depicted in Fig. 4. The input dataset is normalized by the transmit data symbol so that data entries are within the nominal range, potentially aiding the training. We generate 50,000 training samples and 2000 test samples of channel realizations, respectively. The transmit data symbols are modulated using QPSK and 8PSK modulation schemes. The training SINR is obtained from random uniform distribution $\Gamma_{\text{train}} \sim \mathcal{U}(\Gamma_{\text{low}}, \Gamma_{\text{high}})$.

TABLE 5. Inference memory utilization.

Models	Weights	Activations	Memory usage (MB)	Memory saving
SLP-DNet	$(32-\text{bit}) \in \mathbb{R}$	$(32-\text{bit}) \in \mathbb{R}$	0.1898	–
SLP-DBNet	$\{-1,+1\}$	$\{-1,+1\}$	0.0089	21.33×
SLP-DTNet	$\{-1,0,+1\}$	$\{-1,+1\}$	0.0146	13×
SLP-DSQBNet	$\{-\beta_{\text{qf}},\beta_{\text{qf}}\}$	$\{-\beta_{2-bit},\beta_{2-bit}\}$	0.0548	3.46×
SLP-DSQTNet	$\{-\beta_{\text{qf}},0,\beta_{\text{qf}}\}$	$\{-\beta_{2-bit},\beta_{2-bit}\}$	0.0719	2.64×

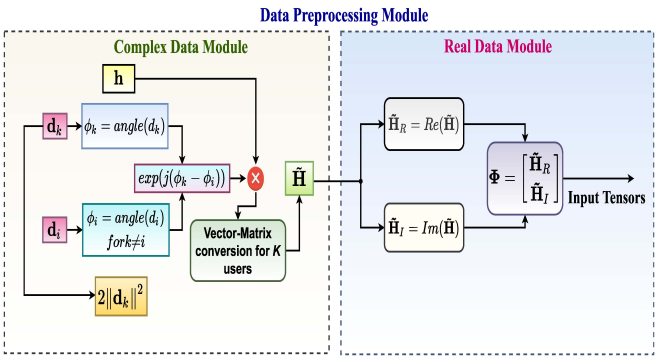


FIGURE 4. Dataset generating and preprocessing Block.

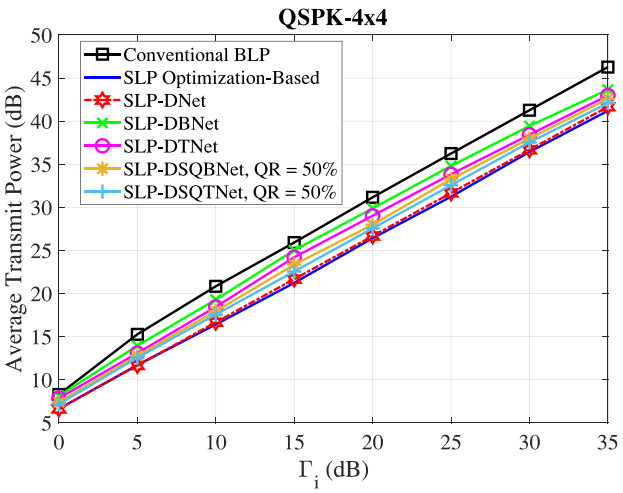


FIGURE 5. Transmit Power vs SINR averaged over 2000 test samples for Conventional Block Level Precoding, SLP optimization-based and nonrobust quantized learning-based SLP solutions, $M = 4$, $K = 4$ and $QR = 50\%$.

From the modulated complex base-band symbols of the i-th user, corresponding real and imaginary channel instantiations are obtained from the relation: $\tilde{\mathbf{H}}_i = \mathbf{H}_i e^{j(\phi_1 - \phi_k)}$, $\tilde{\mathbf{H}}_{Re} = \text{Re}\{\tilde{\mathbf{H}}_i\}$ and $\tilde{\mathbf{H}}_{Re} = \text{Im}\{\tilde{\mathbf{H}}_i\}$. Consequently, the input channel instantiations matrix; $\Phi = \begin{bmatrix}\tilde{\mathbf{H}}_{\text{Re}} & \tilde{\mathbf{H}}_{\text{Im}}\end{bmatrix}^T$ is used to calculate the initial input precoding vectors based on the optimal precoding functions (10) and (15) that will be used to train the model in an unsupervised manner, as shown in the Lagrangian module of Fig. 2. The whole dataset generation process is summarized in Fig. 4.

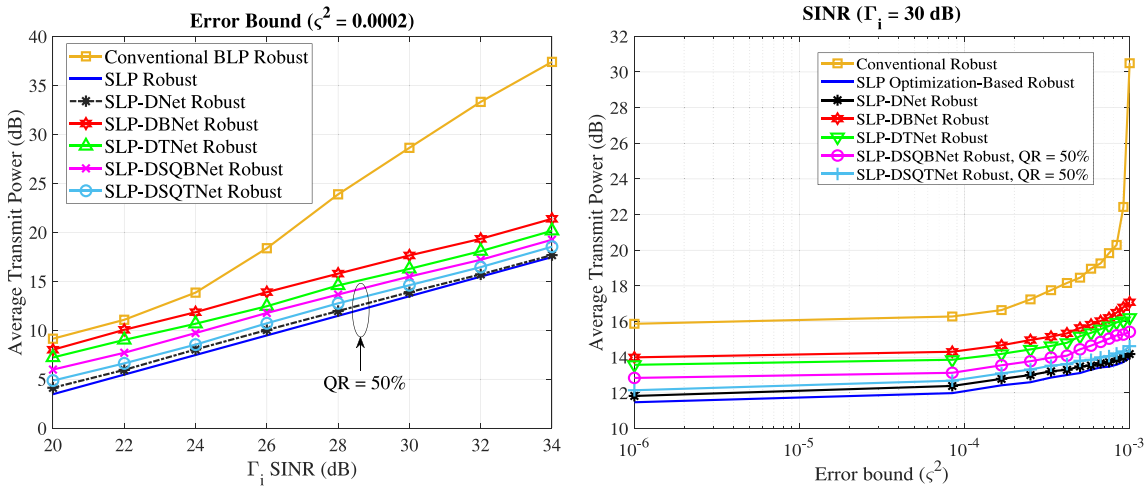
For the training, a stochastic gradient descent (SGD) algorithm is used with the Lagrangian function as a loss metric. A parametric rectified linear unit (**PReLu**) activation function is used for both convolutional and fully connected layers in a full-precision SLP-DNet and the low-bit activation function (24) for SLP-SQDNet. The learning rate is reduced by a factor $\alpha = 0.65$ after every iteration during training to help the learning algorithm converge faster, thereby allowing the network to adapt to the complexity of the problem and enhance its generalization ability. The models are implemented in Pytorch 1.7.1 and Python 3.7.8 on a computer with the following specifications: Intel(R) Core (TM) i7-6700 CPU Core, 32.0GB of RAM. Table 1 summarizes the simulation parameters, while Tables 2 and 3 depict the NN component settings of the SLP-DNet [29].

B. PERFORMANCE EVALUATION OF QSLP-DNET AND SLP-DNET

In the following set of results we compare our proposed quantized DL-based SLP scheme's performance against its corresponding full-precision (SLP-DNet) counterpart's [29] and other benchmark schemes, such as conventional BLP [4], [37] and the classical optimization-based SLP [9], [21]. Primarily, we design full low-bit binary and ternary SLP-DNet models (SLP-DBNet and SLP-DTNet), where the full-precision weights are constrained to 1-bit. Similarly, the expressive learning abilities of SLP-DBNet and SLP-DTNet are further enhanced by designing their corresponding low-bit hybrid stochastically quantized versions (SLP-DSQBNet and SLP-DSQTNet), where part of the weight matrix is quantized to a lower bit, while the remaining is left in its 32-bit floating-point precision. The resulting weight matrix is a hybrid containing both binary and full-precision entries with the activations all reduced to 2-bit according to (24).

The performances of SLP-DBNet, SLP-DTNet, SLP-DSQBNet, SLP-DSQTNet for $QR = 0.5$ against SLP-DNet and other benchmark precoding schemes (conventional BLP, SLP optimization-based) are shown in Fig. 5. It can be observed that both SLP-DBNet and SLP-DTNet have higher transmit power than the SLP optimization-based and SLP-DNet schemes. Therefore, SLP optimization-based and SLP-DNet solutions require less power to transmit the same amount of data symbols than SLP-DBNet and SLP-DTNet. The loss in performance is expected because some information is lost during feed-forward weight/input convolutions due to quantization and the inhomogeneous nature of the quantization errors.

Furthermore, a closer examination of Fig. 5 reveals that the SLP-DSQBNet and SLP-DSQTNet offer less transmit power than their corresponding full binary and ternary versions. Our



(a) Transmit Power vs SINR averaged over 2000 test samples for conventional, SLP optimization-based and robust quantized learning-based SLP solutions under $M = 4$, $K = 4$ and $\varsigma^2 = 0.0002$

(b) Transmit Power vs Error-bound for Conventional BLP, robust SLP optimization-based and robust qunatized learning-based SLP solutions under $M = 4$, $K = 4$ and $QR = 50\%$

FIGURE 6. Performance evaluation of Robust formulation for Conventional BLP, SLP Optimization-based and SLP Learning-based schemes.

simulation also shows that learning by stochastic quantization results in the performance close to the full-precision learning model (SLP-DNet) with a significant model size reduction (memory savings at the inference), as we shall see later. We argue that the decrease in the available transmit power at the BS in this scenario is because not all the weights matrix rows are quantized at once. The quantization error is used to direct the gradient descent towards the best local minima during training. Accordingly, we find that at 30dB, the performance of SLP-DBNet and SLP-DTNet falls by 58% and 35% of the SLP optimization-based solution, respectively. On the other hand, the performance gaps of SLP-DSQBNet, SLP-DSQTNet, and SLP-DNet are 22.2%, 9.62%, and 5% of the SLP optimization-based solution, respectively. Therefore, while the fully quantized model's accuracy is significantly low, the stochastically hybrid quantized counterparts and full-precision models' accuracy is within $88\% - 96\%$ of the optimal solution.

C. PERFORMANCE EVALUATION OF ROBUST SLP-SQDNET

Figs. 6(a) and 6(b) compare the performances of SLP-SQDNet, the conventional and classical CI-based robust precoding schemes for the 4×4 MISO system evaluated at $\varsigma^2 = 2 \times 10^{-4}$. Fig. 6(a) depicts how the average transmit power increases with the *SNR* thresholds, for CSI error bounds $\varsigma^2 = 2 \times 10^{-4}$ and $QR = 50\%$. The robust SLP optimization-based is observed to show a significant power savings of more than 60% compared to the robust conventional BLP. Similarly, the proposed unsupervised learning-based precoders portray similar transmit power reduction trend. They show considerable power savings of $40\% - 58\%$ against the conventional robust BLP. While the fully quantized models have demonstrated substantial performance loss compared to SLP-based optimal precoder, SLP-DSQBNet and SLP-DSQTNet offer $90\% - 98\%$ striking performance correlation with the SLP optimization-based optimal solutions, respectively.

Furthermore, we investigate the effect of the CSI error bounds on the transmit power at 30dB. Fig. 6(b) depicts the variation of the transmit power with increasing CSI error bounds. Moreover, a significant increase in transmit power can be observed where the channel uncertainty lies within the region of CSI error bounds of $\varsigma^2 = 1 \times 10^{-3}$. Interestingly, like the SLP optimization-based algorithm, by exploiting the CI, the proposed unsupervised learning methods also show a descent or moderate increase in transmit power.

D. IMPACT OF QUANTIZATION RATIO

To further understand the impact of the *QR* on the transmit power, Fig. 7 compares the performance of the proposed stochastic quantization learning-based nonrobust precoders evaluated at 30dB and 15dB. Here, we observe that the average transmit power available at the BS required to transmit data symbols increases as more weights and activations are quantized towards extreme quantization to the right ($QR \rightarrow 100$). Fig. 7(a) and Fig. 7(b) show the effect of the quantization ratio on the transmit power at 30dB and 15dB SINR. We observe less power is required to transmit the same amount of symbols at 15 dB SINR. This could be attributed to learning sensitivity at higher SINR as more weights are quantized. As expected, an unquantized SLP-DNet will have more learning ability than the quantized one and performs better, accounting for the relatively less transmission power. We observe that starting with an unquantized SLP-DNet, the

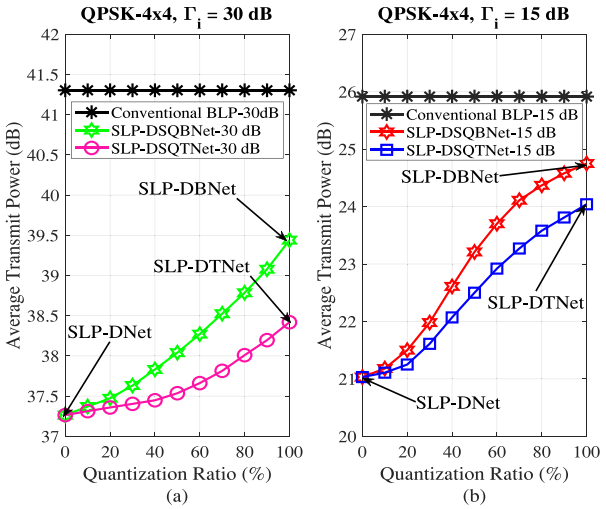


FIGURE 7. Transmit Power vs Quantization ratio averaged over 2000 test samples for nonrobust SQ SLP-DNet models and full-precision SLP-DNet model under $M = 4$, $K = 4$ and $\Gamma = 30$ dB and 15 dB.

transmission power increase gradually as more layer weights are quantized. Fully quantized SLP-DNet models (SLP-DSBNet and SLP-DSTNet) have much lower computational costs than their unquantized counterparts. However, stochastic hybrid quantization could improve the performance loss due to these extreme quantizations, where only a fraction of the weights is quantized to lower bits. For example, in Fig. 7(a), 0.25 dB and 0.75 dB performance margins are observed for 50% binary and ternary stochastic quantizations, respectively. But for fully quantized SLP-DNet, the margins increase to 0.51 dB and 2.25 dB. Nonetheless, we note that even with these trade-offs, SLP-DNet quantized offers more than 3× less transmit power than the conventional BLP.

Fig. 8(a) shows the average transmit power vs quantization ratio (i.e., the proportion of weights that are quantized) at 30dB SINR. The average power at $QR = 0$ corresponds to SLP-DNet while $QR = 1$ represents the corresponding fully quantized counterparts (SLP-DBNet and SLP-DTNet). Moreover, the transmit power gradually increases as more weights are quantized. It is important to note that for a unit quantization ratio ($QR = 1.0$), all the weights are 100% quantized, where the model could be either a typical binary or ternary. On this note, it is clear that the SLP-DSQTNet offers less transmit power than SLP-SQDBNet. We find that quantizing half of the weights ($QR = 50\%$) could guarantee a good performance within $80\% - 98\%$ of the full-precision model for both SLP-SQDBNet and SLP-DSQTNet, respectively. To investigate the amount of the memory required at inference with the increase in the quantization ratio, we plot the model size vs QR as depicted in Fig. 8(b). We find that less memory is required as the quantization moves towards extreme binarization to the right of the QR-axis. It can be seen that the continuous line represents a full-precision SLP-DNet (i.e., $QR = 0$), while $QR = 1$ represents a fully quantized model.

E. COMPLEXITY AND MEMORY EVALUATION

The proposed learning schemes' complexities are examined in two folds: firstly, we compare the number of FLOPs operations involved in our proposed learning methods and those of the benchmark precoding schemes'. Secondly, we evaluate and assess the inference memory requirements of our proposed learning-based precoding techniques.

1) NUMBER OF FLOPS OPERATIONS

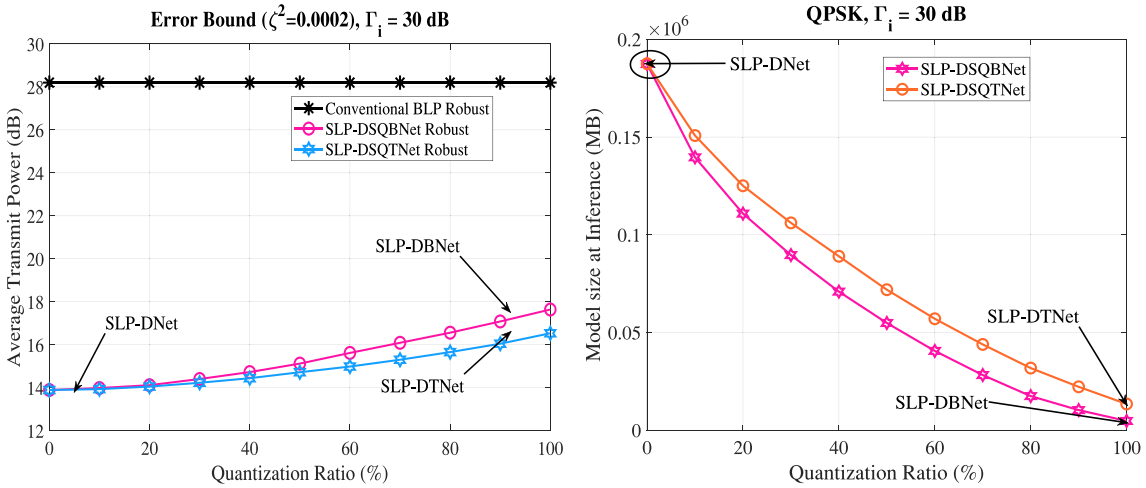
The computational costs of the SLP-DNet are obtained from the PUM and the feed-forward convolutions of the CNN that makes up an APM. For the PUM, the dominant computational cost comes from computing the proximal barrier term [29]. It can be seen that both SLP optimization-based algorithm and the proposed learning schemes are feasible for all sets of M BS antennas and K mobile users. However, for conventional BLP, the solution is only feasible for $M \geq K$.

Fig. 9(a) shows the number of FLOPs operations of the proposed unsupervised learning solutions per symbol for nonrobust formulations. The dominant operations involved in SLP-DNet at the inference are matrix-matrix or vector-matrix convolution. The gap in the computational cost between SLP-DNet and SLP optimization-based methods increases with the growing number of mobile users. For example, we find that the complexity of SLP-DNet is $\sim 10\times$ lower than SLP optimization-based at $K = 10$, while that of SLP-DSQBNet and SLP-DSQTNet are $\sim 20\times$ much lower due to the presence of binary operations. Furthermore, SLP-DBNet and SLP-DTNet offer an additional computational complexity reduction than SLP-DSQBNet and SLP-DSQTNet because binary bit-wise operations replace the entire MAC calculations in the feed-forward pass. It is important to recall that SLP-DTNet outperforms SLP-DBNet in all scenarios. However, we observe that SLP-DTNet is slightly slower than SLP-DBNet, and this is due to the additional '0' binary state introduced in the former. We also note that the advantages of the SLP-DBNet and SLP-DTNet are further enhanced via stochastic quantization but at the expense of small additional complexity overhead. The same trend is also observed in the case of a robust channel scenario, as shown in Fig. 9(b).

Accordingly, we can deduce that while fully binarized DNN could offer significant training and inference accelerations, it could otherwise lead to significant performance degradation. However, quantizing the weight matrix via a stochastic channel selection based on the quantization error leads to the improved performance (lower transmission power) comparable to the classical SLP optimization-based solution. Therefore, we can conclude that the results in Figs. 9(a) and 9(b) demonstrate that the proposed quantized DL-based SLP solutions offer a good trade-off between the performance and computational complexity.

2) MODEL SIZE AND MEMORY UTILIZATION

Generally, GPU can speedup the offline training of DNNs. However, most modern GPUs are memory-constrained (e.g., GTX 980: 4GB, Tesla K40: 12GB, Tesla K20: 5GB and



(a) Transmit Power vs Quantization Ratio averaged over 2000 test samples for SLP optimization-based and SQ SLP-DNet models for robust formulations, $M = 4$, $K = 4$ and $\Gamma_i = 30$ dB

(b) Memory requirement at Inference vs Quantization ratio for SLP-based Full-precision SLP-DNet, SLP-DSQBNet and SLP-DSQTNet under $M = 4$, $K = 4$ and $\Gamma_i = 30$dB

FIGURE 8. Average power and inference memory requirement vs quantization error of the proposed learning-based precoding schemes.

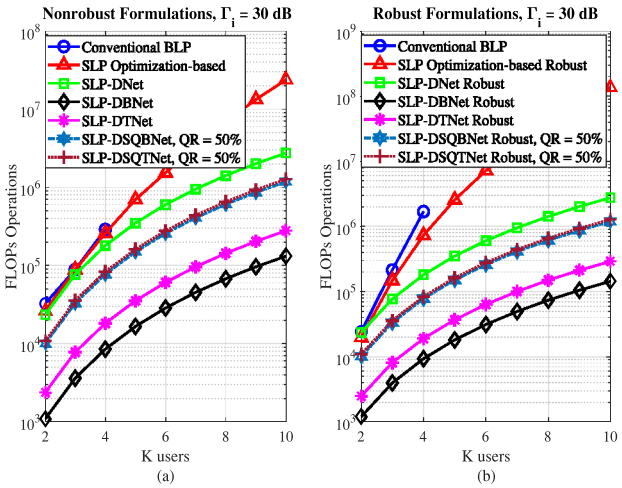


FIGURE 9. Comparison of FLOPs operations performed for Nonrobust and Robust precoding schemes, i.e., conventional BLP, SLP optimization-based and SLP learning-based models using four BS antennas ($M = 4$) and $QR = 50\%$.

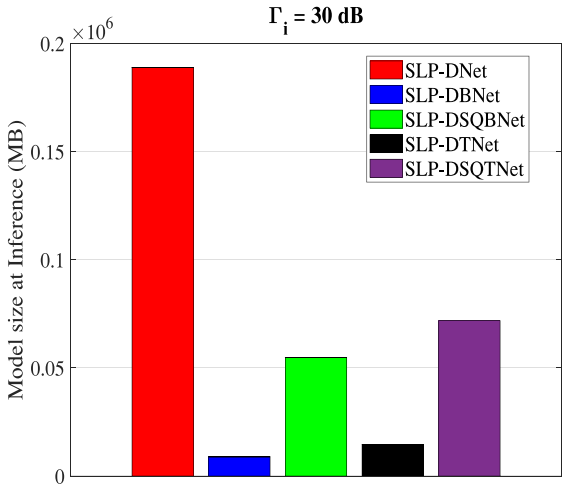


FIGURE 10. Memory requirement at Inference for Full-precision SLP-DNet, SLP-DBNet, SLP-DTNet, SLP-DSQBNet and SLP-DSQTNet under $M = 4$, $K = 4$, $\Gamma_I = 30$dB and $QR = 50\%$.

GTX Titan X: 12GB) [46]. Practically, the size of the DNN is often bounded by the available memory. Therefore, it is beneficial to estimate the memory requirements of the DNN at the inference. Likewise, the actual memory utilization also depends on the implementation. Here, we examine and analyze the memory utilization of full-precision SLP-DNet and its corresponding quantized versions at inference. By memory utilization, we refer to the model size at the testing phase. For this analysis, we adopt the approach presented in [47] to calculate the inference memory utilization as the summation of 32-bit times the number of floating-point parameters and 1-bit times the number of binary parameters. Mathematically, this can be expressed as $\frac{1}{32}W_b + W_f$, where W_b and W_f are the binary and floating-point weights, respectively.

Furthermore, Fig. 10 shows that SLP-DBNet and SLP-DBNet provide considerable memory savings up to $\sim 21\times$ and $\sim 13\times$ compared to the full-precision SLP-DNet because the extreme quantization reduces the available learning parameters significantly. This brings about a trade-off between performance and model size, which is compensated by hybrid quantization as in SLP-DSQBNet and SLP-DSQTNet.

Fig. 11 shows memory efficiency/computational complexity vs. performance. It can be observed that the computational complexity is proportional to the size of the network. As more weights are quantized, the network's size decreases, and more power is required for transmission. Interestingly, SLP-DSQTNet almost matches the performance of the full-precision SLP-DNet with $2.64\times$ memory savings. Similarly,

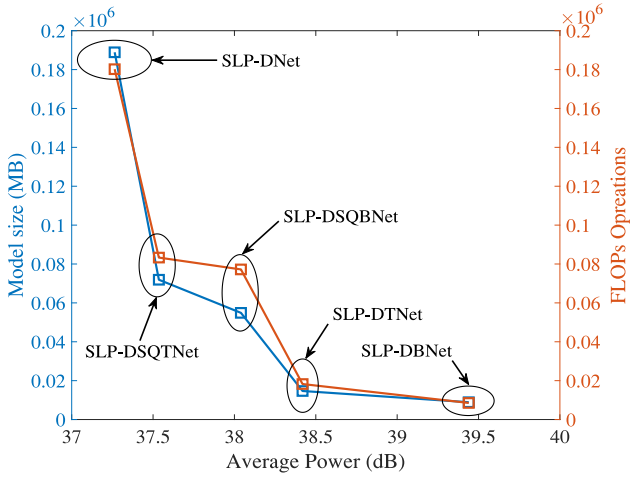


FIGURE 11. Memory requirement, Complexity vs Average transmit Power for SLP-DNet, SLP-DBNet, SLP-DTNet, SLP-DSQBNet and SLP-DSQTNet under $M = 4$, $K = 4$, $\Gamma_i = 30$dB and $QR = 50\%$.

SLP-DSQBNet yields 3.46× memory savings at 0.5 dB performance loss against the SLP-DNet. Similarly, more memory is further saved in the case of extreme quantization. This is particularly important when deploying the trained model to resource-constrained edge devices, where memory or space is critical. In such a case, our proposed approach offers a scalable trade-off between complexity and memory efficiency. For instance, if power efficiency (PE) in terms of BS's power and the processing power of the SLP schemes is the main priority, our proposed approach could be tuned to match classical SLP scheme. However, if memory is a priory, one can observe that the proposed SLP-DSQBNet and SLP-DSQTNet perform much better than both classical SLP optimization-based and full-precision SLP-DTNet techniques. Moreover, SLP-DSQBNet, SLP-DSQTNet, SLP-DBNet, and SLP-DTNet schemes are much more PE than BLP method.

Table 5 presents the summary of the inference memory requirements, MAC, and binary operations of different proposed learning implementations. For SLP-DSQBNet and SLP-DSQTNet, the weights are constrained to the following quantization $\{-\beta_{\text{qf}}, \beta_{\text{qf}}\}$ and $\{-\beta_{\text{qf}}, 0, \beta_{\text{qf}}\}$ while the activations are clipped to $\{-\beta_{2-bit}, \beta_{2-bit}\}$ $2-$bit quantized values, respectively. This shows that the hybrid quantization enhances the representational capabilities of the convolutional block.

VI. CONCLUSION

This paper proposed a hybrid quantization DNN-based SLP scheme termed (SLP-QSDNet) based on binary and ternary operations for power minimization for a multi-user downlink MISO system. We proposed various weight quantization techniques to obtain its corresponding full and partially quantized counterparts. We showed that the proposed approach resulted in fast online learning and a significant model size reduction, which could help render the trained model memory-efficient during deployment on the device's edge. Overall, our proposed approaches provide a scalable trade-off between performance and complexity in learning-based SLP transmission.

REFERENCES

[1] C. Windpassinger, R. F. H. Fischer, T. Vencel, and J. B. Huber, "Precoding in multiantenna and multiuser communications," *IEEE Trans. Wireless Commun.*, vol. 3, no. 4, pp. 1305–1316, Jul. 2004.

[2] V. Stankovic and M. Haardt, "Multi-user MIMO downlink precoding for users with multiple antennas," in *Proc. 12th Meeting Wireless World Res. Forum (WWRF)*, vol. 10. Toronto, ON, Canada, 2004, pp. 12–14.

[3] J. Lee and N. Jindal, "High SNR analysis for MIMO broadcast channels: Dirty paper coding versus linear precoding," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4787–4792, Dec. 2007.

[4] E. Björnson, M. Bengtsson, and B. Ottersten, "Optimal multiuser transmit beamforming: A difficult problem with a simple solution structure [lecture notes]," *IEEE Signal Process. Mag.*, vol. 31, no. 4, pp. 142–148, Jul. 2014.

[5] C. Masouros and E. Alsusa, "A novel transmitter-based selective-precoding technique for DS/CDMA systems," in *Proc. IEEE Int. Conf. Commun.*, 2007, pp. 2829–2834.

[6] C. Masouros, "Correlation rotation linear precoding for MIMO broadcast communications," *IEEE Trans. Signal Process.*, vol. 59, no. 1, pp. 252–262, Jan. 2011.

[7] C. Masouros, M. Sellathurai, and T. Ratnarajah, "Vector perturbation based on symbol scaling for limited feedback MISO downlinks," *IEEE Trans. Signal Process.*, vol. 62, no. 3, pp. 562–571, Feb. 2014.

[8] M. Alodeh, S. Chatzinotas, and B. Ottersten, "Constructive multiuser interference in symbol level precoding for the MISO downlink channel," *IEEE Trans. Signal Process.*, vol. 63, no. 9, pp. 2239–2252, May 2015.

[9] C. Masouros and G. Zheng, "Exploiting known interference as green signal power for downlink beamforming optimization," *IEEE Trans. Signal Process.*, vol. 63, no. 14, pp. 3628–3640, Jul. 2015.

[10] P. V. Amadori and C. Masouros, "Constant envelope precoding by interference exploitation in phase shift keying-modulated multiuser transmission," *IEEE Trans. Wireless Commun.*, vol. 16, no. 1, pp. 538–550, Jan. 2017.

[11] D. Spano, M. Alodeh, S. Chatzinotas, and B. Ottersten, "Symbol-level precoding for the nonlinear multiuser MISO downlink channel," *IEEE Trans. Signal Process.*, vol. 66, no. 5, pp. 1331–1345, Mar. 2018.

[12] C. Masouros, "Harvesting signal power from constructive interference in multiuser downlinks," in *Wireless Information and Power Transfer: A New Paradigm for Green Communications*. Cham, Switzerland: Springer, 2018, pp. 87–122.

[13] A. Li, C. Masouros, F. Liu, and A. L. Swindlehurst, "Massive MIMO 1-bit DAC transmission: A low-complexity symbol scaling approach," *IEEE Trans. Wireless Commun.*, vol. 17, no. 11, pp. 7559–7575, Nov. 2018.

[14] M. R. A. Khandaker, C. Masouros, and K.-K. Wong, "Constructive interference based secure precoding: A new dimension in physical layer security," *IEEE Trans. Inf. Forensics Security*, vol. 13, pp. 2256–2268, 2018.

[15] A. Li, C. Masouros, B. Vucetic, Y. Li, and A. L. Swindlehurst, "Interference exploitation precoding for multi-level modulations: Closed-form solutions," *IEEE Trans. Commun.*, vol. 69, no. 1, pp. 291–308, Jan. 2021.

[16] A. Li and C. Masouros, "A two-stage vector perturbation scheme for adaptive modulation in downlink MU-MIMO," *IEEE Trans. Veh. Technol.*, vol. 65, no. 9, pp. 7785–7791, Sep. 2016.

[17] A. Li and C. Masouros, "Exploiting constructive mutual coupling in P2P MIMO by analog-digital phase alignment," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1948–1962, Mar. 2017.

[18] K. L. Law, C. Masouros, and M. Pesavento, "Transmit precoding for interference exploitation in the underlay cognitive radio Z-channel," *IEEE Trans. Signal Process.*, vol. 65, no. 14, pp. 3617–3631, Jul. 2017.

[19] A. Li and C. Masouros, "Hybrid massive MIMO unlicensed transmission with 1-bit quantization," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, 2017, pp. 1–6.

[20] A. Li and C. Masouros, "Interference exploitation precoding made practical: Optimal closed-form solutions for PSK modulations," *IEEE Trans. Wireless Commun.*, vol. 17, no. 11, pp. 7661–7676, Sep. 2018.

[21] A. Li et al., "A tutorial on interference exploitation via symbol-level precoding: Overview, state-of-the-art and future directions," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 796–839, 2nd Quart., 2020.

[22] Z. Xiao, R. Liu, M. Li, Y. Liu, and Q. Liu, "Low-complexity designs of symbol-level precoding for MU-MISO systems," *IEEE Trans. Commun.*, vol. 70, no. 7, pp. 4624–4639, Jul. 2022.

[23] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2554–2564, May 2019.

[24] A. Mohammad, C. Masouros, and Y. Andreopoulos, "Complexity-scalable neural-network-based MIMO detection with learnable weight scaling," *IEEE Trans. Commun.*, vol. 68, no. 10, pp. 6101–6113, Oct. 2020.

[25] L. Liang, H. Ye, G. Yu, and G. Y. Li, "Deep-learning-based wireless resource allocation with application to vehicular networks," *Proc. IEEE*, vol. 108, no. 2, pp. 341–356, Feb. 2020.

[26] A. Alkhateeb, S. Alex, P. Varkey, Y. Li, Q. Qu, and D. Tujkovic, "Deep learning coordinated beamforming for highly-mobile millimeter wave systems," *IEEE Access*, vol. 6, pp. 37328–37348, 2018.

[27] W. Xia, G. Zheng, Y. Zhu, J. Zhang, J. Wang, and A. P. Petropulu, "A deep learning framework for optimization of MISO downlink beamforming," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1866–1880, Mar. 2020.

[28] P. de Kerret and D. Gesbert, "Robust decentralized joint precoding using team deep neural network," in *Proc. 15th Int. Symp. Wireless Commun. Syst. (ISWCS)*, 2018, pp. 1–5.

[29] A. Mohammad, C. Masouros, and Y. Andreopoulos, "An unsupervised learning-based approach for symbol-level-precoding," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2021, pp. 1–6.

[30] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 6869–6898, 2017.

[31] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 525–542.

[32] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1398–1406.

[33] Y. Dong, R. Ni, J. Li, Y. Chen, H. Su, and J. Zhu, "Stochastic quantization for learning accurate low-bit deep neural networks," *Int. J. Comput. Vis.*, vol. 127, no. 11, pp. 1629–1642, 2019.

[34] C. Masouros and T. Ratnarajah, "Interference as a source of green signal power in cognitive relay assisted co-existing MIMO wireless transmissions," *IEEE Trans. Commun.*, vol. 60, no. 2, pp. 525–536, Feb. 2012.

[35] C. Masouros, M. Sellathurai, and T. Ratnarajah, "Large-scale MIMO transmitters in fixed physical spaces: The effect of transmit correlation and mutual coupling," *IEEE Trans. Commun.*, vol. 61, no. 7, pp. 2794–2804, Jul. 2013.

[36] R. Hauser, *Lecture 10, Numerical Linear Algebra and Optimisation Oxford University Computing Laboratory: "Interior-Point Methods for Inequality Constrained Optimisation"*. Oxford, U.K.: Oxford Univ., 2007. [Online]. Available: https://people.maths.ox.ac.uk/hauser/hauser_lecture4.pdf

[37] G. Zheng, K.-K. Wong, and T.-S. Ng, "Robust linear MIMO in the downlink: A worst-case optimization with ellipsoidal uncertainty regions," *EURASIP J. Adv. Signal Process.*, vol. 2008, pp. 1–15, Jul. 2008.

[38] A. Mohammad, C. Masouros, and Y. Andreopoulos, "Accelerated learning-based MIMO detection through weighted neural network design," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–6.

[39] T. Abtahi, A. Kulkarni, and T. Mohsenin, "Accelerating convolutional neural network with FFT on Tiny cores," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2017, pp. 1–4.

[40] M. Courbariaux, Y. Bengio, and J.-P. David, "BinaryConnect: Training deep neural networks with binary weights during propagations," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, vol. 2, 2015, pp. 3123–3131.

[41] H. Alemdar, V. Leroy, A. Prost-Boucle, and F. Pétrot, "Ternary neural networks for resource-efficient AI applications," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2017, pp. 2547–2554.

[42] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6848–6856.

[43] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[44] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Proc. 30th Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 4114–4122.

[45] K.-Y. Wang, A. M.-C. So, T.-H. Chang, W.-K. Ma, and C.-Y. Chi, "Outage constrained robust transmit optimization for multiuser MISO downlinks: Tractable approximations by conic optimization," *IEEE Trans. Signal Process.*, vol. 62, no. 21, pp. 5690–5705, Nov. 2014.

[46] P. Vingelmann and F. H. Fitzek. "CUDA, release: 10.2.89." NVIDIA. 2020. [Online]. Available: https://developer.nvidia.com/cuda-toolkit

[47] J. Bethge, H. Yang, M. Bornstein, and C. Meinel, "BinaryDenseNet: Developing an architecture for binary neural networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops*, 2019, pp. 1951–1960.

ABDULLAHI MOHAMMAD (Student Member, IEEE) received the B.Eng. and M.Sc. degrees in electrical engineering from Ahmadu Bello University, Zaria, Nigeria, in 2008 and 2014, respectively, and the Ph.D. degree in electronic and electrical engineering from University College London, U.K, in 2022. He is currently a Lecturer with the Department of Computer Engineering, Ahmadu Bello University. His main research interests include machine learning for wireless communications, information theory, beamforming for MIMO communication systems, RF fingerprinting for physical layer security, and control systems. He is a fully Registered Engineer with the Council for the Regulation of Engineering in Nigeria (COREN), a regulatory body that governs the practice of engineering in Nigeria.

CHRISTOS MASOUROS (Senior Member, IEEE) received the Diploma degree in electrical and computer engineering from the University of Patras, Greece, in 2004, and the M.Sc. degree by research and the Ph.D. degree in electrical and electronic engineering from The University of Manchester, U.K., in 2006 and 2009, respectively. In 2008, he was a Research Intern with Philips Research Labs, U.K. From 2009 to 2010, he was a Research Associate with The University of Manchester. From 2010 to 2012, he was a Research Fellow with Queen's University Belfast. In 2012, he joined University College London as a Lecturer, where he is currently a Full Professor with the Information and Communications Engineering Research Group, Department of Electrical and Electronic Engineering. He has held a Royal Academy of Engineering Research Fellowship from 2011 to 2016. His research interests include wireless communications and signal processing with a particular focus on green communications, large-scale antenna systems, cognitive radio, interference mitigation techniques for MIMO, and multi-carrier communications. He was a recipient of the Best Paper Awards at IEEE GlobeCom 2015 and IEEE WCNC 2019 conferences. He is currently the Chair of the IEEE Special Interest Group on Energy Harvesting and an Elected Member of the EURASIP SAT Committee on Signal Processing for Communications and Networking. He has been recognized as an Exemplary Editor of IEEE COMMUNICATIONS LETTERS and an Exemplary Reviewer of IEEE TRANSACTIONS ON COMMUNICATIONS. He is an Editor of IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, and IEEE OPEN JOURNAL OF SIGNAL PROCESSING and the Editor-at-Large of IEEE Open Journal of the Communications Society. He has been an Associate Editor of IEEE COMMUNICATIONS LETTERS and a Guest Editor of IEEE JOURNAL ON SELECTED TOPICS IN SIGNAL PROCESSING issues: Exploiting Interference towards Energy Efficient and Secure Wireless Communications and Hybrid Analog/Digital Signal Processing for Hardware-Efficient Large Scale Antenna Arrays. He is a member of IET.

YIANNIS ANDREOPOULOS (Senior Member, IEEE) received the Diploma degree in electrical engineering and the M.Sc. degree from the University of Patras, Greece, and the Ph.D. degree in applied sciences from Vrije Universiteit Brussel, Belgium. He is currently a Professor of Data and Signal Processing Systems with the Electronic and Electrical Engineering Department, University College London, London, U.K. His research interests include machine learning and multimedia systems. In his academic work, he has made major contributions to image/vision computing and large-scale multidimensional data analysis. In collaboration with the industry, elements of his research in video processing systems have been integrated into commercial products.