Titre: Title:	Compression of Site-Specific Deep Neural Networks for Massive MIMO Precoding
Auteur: Author:	Ghazal Kasalaee
Date:	2025
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Kasalaee, G. (2025). Compression of Site-Specific Deep Neural Networks for Massive MIMO Precoding [Master's thesis, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/68392/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/68392/
Directeurs de recherche: Advisors:	Jean-François Frigon, & François Leduc-Primeau
Programme: Program:	Génie électrique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Compression of Site-Specific Deep Neural Networks for Massive MIMO Precoding

GHAZAL KASALAEE
Département de génie électrique

Mémoire présenté en vue de l'obtention du diplôme de *Maître ès sciences*
Génie électrique

Août 2025

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Compression of Site-Specific Deep Neural Networks for Massive MIMO Precoding

présentée par **Ghazal KASALAEE**
en vue de l'obtention du diplôme de *Maître ès sciences*
a été dûment acceptée par le jury d'examen constitué de :

Gunes KARABULUT KURT, présidente
Jean-François FRIGON, membre et directeur de recherche
François LEDUC-PRIMEAU, membre et codirecteur de recherche
Tri Nhu DO, membre

DEDICATION

To my cherished mother, Mahnaz, and my beloved father, Afshin — With deepest gratitude and love....

ACKNOWLEDGEMENTS

I would like to extend my deepest gratitude to all those whose support and encouragement have been vital to the completion of this thesis.

Above all, I am sincerely thankful to my supervisors, Professor François Leduc-Primeau and Professor Jean-François Frigon, for their exceptional guidance, expertise, and unwavering support throughout this journey. Their mentorship has played a crucial role in shaping my research and nurturing my academic growth. I would like to express my sincere gratitude to doctoral fellow Ali Hasanzadeh Karkan for his invaluable contributions as a co-author of our paper. His insightful guidance, constructive feedback, and dedication were instrumental to the development of this work.

I am especially grateful to my dear mother, Mahnaz, and my beloved father, Afshin, whose unconditional love, steadfast encouragement, and countless sacrifices have laid the foundation for every step of my journey. Their unwavering belief in me has been a constant source of strength and inspiration.

RÉSUMÉ

La migration vers les réseaux sans fil de cinquième génération (5G) et au-delà impose une contrainte énergétique inédite aux stations de base, en particulier pour le précodage à très grand nombre d'antennes (MIMO massif ou mMIMO) dont la charge de calcul croît fortement avec le nombre d'antennes. Ce mémoire propose des précodages fondés sur l'apprentissage profond qui maximisent le rendement énergétique tout en conservant une excellente efficacité spectrale.

La première partie démontre qu'une compression de modèles spécifiques au site grâce à un entraînement avec quantification en précision mixte combiné à la recherche d'architectures neuronales (NAS) réduit l'énergie de calcul jusqu'à 35 fois par rapport au précodage de référence à erreur quadratique moyenne pondérée (WMMSE) pour un débit identique. Le processus s'appuie sur un modèle de consommation d'énergie calibré au matériel qui intègre les opérations multiplication-accumulation et les accès mémoire sur puce, révélant des compromis de Pareto sur un jeu de canaux simulés par lancer de rayons, incluant des environnements urbains en visée directe et indirecte.

La seconde partie introduit BitAdapt, une méthode entièrement dérivable d'apprentissage de la précision qui évite la recherche combinatoire des résolutions de quantification. Chaque couche possède une résolution apprise couplée à la méthode de quantification apprise LSQ, tandis que le modèle d'énergie est intégré directement dans la fonction de coût. L'optimisation conjointe des poids, des pas de quantification et des résolutions place chaque couche au point de compromis optimal entre précision et énergie, générant des configurations en précision mixte adaptées aux conditions de canal propres à chaque cellule. Appliqué à un réseau convolutif léger comme à un réseau plus profond de type Transformer, BitAdapt réduit d'un ordre de grandeur l'énergie d'inférence par rapport à une quantification uniforme sur 8-bit, tout en égalant, voire dépassant, le débit total multi-utilisateur obtenu avec WMMSE.

Ces contributions ouvrent la voie à des précodages neuronaux sobres en énergie pour les stations de base mMIMO, conciliant la flexibilité de l'apprentissage profond et les contraintes de puissance strictes des futures infrastructures sans fil.

ABSTRACT

The push toward 5G-and-beyond wireless networks is placing unprecedented pressure on base station energy budgets, particularly for massive Multiple-Input Multiple-Output (mMIMO) precoding, where computational cost scales steeply with antenna count. This thesis addresses that challenge by optimizing deep-learning-based precoders to maximize energy efficiency while maintaining state-of-the-art spectral efficiency.

The first part of this work demonstrates that careful site-specific compression of a convolutional precoder combining Mixed-Precision Quantization-Aware training with Neural Architecture Search can cut computational energy by up to 35× relative to a Weighted Minimum Mean-Square Error (WMMSE) baseline at equal sum rate. The search is guided by a hardware-calibrated cost model that accounts for both multiply-accumulate operations and on-chip memory accesses, revealing Pareto-optimal trade-offs over a ray-traced data set spanning both line-of-sight and non-line-of-sight urban deployments.

Building on these results, the second part introduces BitAdapt, a fully differentiable precision learning framework that eliminates the need for an outer combinatorial bit-width search. BitAdapt treats each layer's bit-width as a learnable parameter, couples it with a Learned Step Size Quantization (LSQ) scheme, and embeds the same analytic energy surrogate directly into the training objective. By jointly optimizing network weights, quantizer step sizes, and bit-widths, BitAdapt drives every layer to operate at the knee of its accuracy–energy curve, yielding mixed-precision configurations tailored to each cell's channel conditions. Applied to both a lightweight convolutional precoder and a deeper Transformer model-based precoder, BitAdapt delivers an order-of-magnitude reduction in inference energy relative to a uniform 8-bit baseline while preserving and in some cases exceeding the sum-rate performance achieved by WMMSE.

Taken together, these contributions establish a principled, hardware-aware pathway for deploying ultra-efficient neural precoders in mMIMO base stations, effectively bridging the gap between deep learning's adaptability and the strict power constraints of next-generation wireless infrastructure.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

AI	Artificial Intelligence.
ANN	Artificial Neural Network.
BS	Base Station.
CNN	Convolution Neural Network.
CSI	Channel State Information.
DAC	Dynamic Algorithm Configuration.
DL	Deep Learning.
DNN	Deep Neural Network.
DRAM	Dynamic Random Access Memory.
DSP	Digital Signal Processing.
EE	Energy Efficiency.
FCL	Fully Connected Layer.
FDP	Fully Digital Precoder.
FP32	Floating-Point Single Precision.
FPGA	Field-Programmable Gate Array.
IOT	Internet-Of-Things.
KD	Knowledge Distillation.
LOS	Line-Of-Sight.
LR	Learning Rate.
LSQ	Learned Step Size Quantization.
LSTM	Long Short-Term Memory.
MAC	Multiply-Accumulate.
ML	Machine Learning.

mm-Wave	Millimeter Wave.
mMIMO	Massive Multiple-Input Multiple-Output.
MPQ	Mixed-Precision Quantization.
MRT	Maximum-Ratio Transmission.
NAS	Neural Architecture Search.
NLOS	Non-Line-Of-Sight.
NLP	Natural Language Processing.
pJ	PicoJoule.
PTQ	Post-Training Quantization.
QAT	Quantization-Aware Training.
ReLU	Rectified Linear Unit.
RF	Radio Frequency.
RL	Reinforcement Learning.
RNN	Recurrent Neural Networks.
SE	Spectral Efficiency.
SINR	Signal-To-Interference-Noise Ratio.
SNR	Signal-To-Noise Ratio.
SR	Sum-Rate.
SRAM	Static Random Access Memory.
STE	Straight Through Estimator.
WMMSE	Weighted Minimum Mean-Square Error.
ZF	Zero Forcing.

CHAPTER 1 INTRODUCTION

1.1 Context

Over the past decade, mobile communication has transformed nearly every facet of modern life from streaming high-definition video to enabling intelligent transportation and remote healthcare. This rapid growth in connected devices and bandwidth-intensive applications has driven an exponential surge in mobile data traffic. To meet this insatiable demand, the wireless community has pursued ambitious advances embodied in fifth-generation (5G) networks and the emerging vision of sixth-generation (6G) systems. These new generations promise not only unprecedented data rates but also ultra-reliable low-latency communication and the capacity to connect massive numbers of devices.

Yet, alongside these capabilities comes a pressing challenge: Energy Efficiency (EE). As mobile networks grow denser and data rates increase, energy consumption has become a critical bottleneck, both economically and environmentally. Recent estimates indicate that base stations alone account for up to 70% of the total energy consumption in cellular networks [3]. Within each Base Station (BS), significant power is consumed by the baseband processing unit (BBU), particularly in the physical layer tasks of channel estimation, beamforming, and precoding [4].

One of the most powerful technologies driving improvements in spectral efficiency and EE is Massive Multiple-Input Multiple-Output (mMIMO). By deploying large antenna arrays at BS, mMIMO enables simultaneous service to multiple users over the same time-frequency resources, dramatically boosting capacity and link reliability [5]. However, these gains come at a cost: the computational complexity of mMIMO signal processing, especially in downlink precoding, scales rapidly with the number of antennas and users. For instance, practical deployments featuring 64 antennas serving dozens of users require the base station to compute precoding matrices within tight millisecond-level deadlines. This imposes a significant burden on processing hardware and substantially contributes to energy consumption and system latency [6].

Traditional optimization-based precoding methods such as Zero Forcing (ZF) and Weighted Minimum Mean-Square Error (WMMSE) deliver excellent performance but rely on computationally intensive matrix operations, including large matrix inversions. These procedures often prove impractical for real-time deployment under stringent energy and latency constraints [7]. This reality has sparked interest in exploring alternative solutions that can

maintain precoding performance while reducing computational demands.

In recent years, Deep Learning (DL) has emerged as a promising tool in wireless communication, capable of approximating complex, high-dimensional functions that traditional algorithms struggle to handle. Specifically, Deep Neural Networks (DNNs) can be trained to learn the relationship between Channel State Information (CSI) and optimal precoding matrices, offering the potential for rapid, one-shot inference instead of iterative optimization [8]. This aligns well with the shift toward hardware accelerators such as GPUs, FPGAs, and custom ASICs in modern BS, where efficient inference is increasingly feasible.

However, deploying DNNs for precoding introduces new challenges. State-of-the-art networks can involve millions of parameters and require billions of arithmetic operations per inference [9], undermining the gains from eliminating iterative algorithms. Moreover, different neural network architectures exhibit varied trade-offs in latency, power consumption, and hardware compatibility. Consequently, there is a crucial need for a systematic, hardware-aware design of compact and efficient neural networks suitable for real-time wireless applications.

A key technique for reducing the computational footprint of DNNs is quantization, which involves representing network weights and activations using lower bit-widths (e.g., 8-bit, 4-bit, or even binary). While quantization offers significant gains in energy and memory efficiency, it often comes at the cost of reduced model accuracy [10]. Approaches like Quantization-Aware Training (QAT) help mitigate this trade-off by integrating quantization into the training process, allowing the model to adapt to the reduced precision. Additionally, Mixed-Precision Quantization (MPQ) strategies assign different bit-widths to different layers, improving overall efficiency while preserving critical model performance [11]. Yet determining optimal precision allocation remains a complex optimization problem.

Recent innovations such as BitPruning [12] further advance this field by enabling differentiable, gradient-based optimization of bit-widths, jointly learning network parameters and precision levels under energy constraints. In parallel, the variability of wireless environments imposes another challenge: a network trained on generic data may fail to generalize to real deployment scenarios. Site-specific adaptation leveraging ray-traced channel models to capture realistic propagation effects has thus emerged as a crucial step for ensuring robust, deployable solutions [13].

1.2 Specific Objectives

This thesis aims to bridge the gap between the theoretical promise of DL-based precoding and its practical realization in next-generation wireless networks. The overarching objective is to develop an energy-efficient DL framework for downlink precoding in mMIMO systems, capable of delivering high performance within the strict computational and power budgets of real-world BS.

1.3 Thesis Outline

This thesis is structured into five chapters, each building toward a green, deployment-ready DL framework for mMIMO precoding:

- **Chapter 1 – Introduction**
  Presents the context, motivation, and research objectives, emphasizing the need for low-complexity, energy-aware precoding in next-generation wireless networks.
- **Chapter 2 – Background and Literature Review**
  Surveys on mMIMO signal processing, DL in wireless communications, EE metrics, quantization techniques, and Neural Architecture Search (NAS), laying the groundwork for the proposed approach.
- **Chapter 3 – Compression of Site-Specific Deep Neural Networks for mMIMO Precoding** (Based on our published paper [14])
  Presents a QAT-MPQ, NAS-guided compression pipeline and benchmarks its EE against classical precoders such as ZF and WMMSE.
- **Chapter 4 – *BitAdapt*: An Energy-Aware Precision Learning Framework**
  Details the BitAdapt algorithm for differentiable bit-width optimization. including its energy model, training protocol, and architecture-specific implementation for Convolutional Neural Network (CNN) and Transformer precoders. Provides an extensive experimental evaluation across multiple ray-traced deployment sites, analyzing energy–throughput trade-offs, hyper-parameter sensitivity, and layer-wise precision allocation.
- **Chapter 5 – Conclusion and Future Work**
  Summarizes the key findings, discusses current limitations, and outlines directions for extending the framework to emerging 6G scenarios and hardware platforms.

Key Contributions

The main contributions of this thesis are summarized as follows:

- Developing QAT techniques based on Learned Step Size Quantization (LSQ) to preserve precoding performance under reduced numerical precision.
- Employing MPQ strategies, guided by NAS, to optimize per-layer bit-widths for low energy inference.
- Integrating BitPruning for differentiable, end-to-end optimization of bit allocations under hardware-aware energy constraints.
- Proposing an analytic energy model that captures both computational and memory costs of neural precoding, expressed in (bit/s/Hz/ μJ).
- Incorporating site-specific training using ray-traced channel data to enhance the generalization and deployment readiness of the neural precoder.

Through these contributions, this thesis seeks to deliver not only theoretical insights but also practical pathways toward realizing efficient, high-performance neural precoding solutions for future wireless communication systems.

CHAPTER 2 BACKGROUND AND LITERATURE REVIEW

2.1 Introduction to mMIMO

mMIMO systems represent a revolutionary advancement in wireless communication, enabling high Spectral Efficiency (SE), EE, and reliability through the deployment of large-scale antenna arrays. Introduced as an evolution of traditional MIMO, which appeared in the 1970s to enhance capacity and robustness by exploiting multipath propagation, mMIMO extends the concept by scaling the number of antennas at the BS to the order of tens or hundreds [15]. This scaling unlocks new performance boundaries by simultaneously serving multiple users using the same time-frequency resources.

The core principle of mMIMO lies in the exploitation of spatial multiplexing, array gain, and diversity gain. Spatial multiplexing allows parallel transmission of multiple data streams, improving throughput. Array gain enhances received signal strength through coherent beamforming. Diversity gain combats fading by leveraging multiple independent signal paths. These capabilities significantly improve SE, coverage, and user experience.

In mMIMO systems, signals are spatially separated using beamforming techniques. Beamforming adjusts the phase and amplitude of signals across antennas, allowing the formation of narrow, directional beams that target specific users. This helps mitigate inter-user interference and increases Signal-To-Interference-Noise Ratio (SINR). Figure 2.1 illustrates a typical downlink operation where an M-antenna BS transmits distinct data streams to K users.

Let $\mathbf{d} \in \mathbb{C}^{K\times 1}$ denote the user data vector, and $\mathbf{W} \in \mathbb{C}^{N_t\times K}$ the precoding matrix. The BS generates the transmit signal

$$\mathbf{s} = \mathbf{W}\mathbf{d}\,. \tag{2.1}$$

The received signal is then

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n}\,, \tag{2.2}$$

where $\mathbf{H} \in \mathbb{C}^{K\times M}$ denotes the downlink channel matrix, whose k-th row $\mathbf{h}_k^{\mathrm{H}}$ captures the propagation paths from all M transmit antennas to user k, and $\mathbf{n}$ is an additive white Gaussian noise vector. Precoding is a signal processing technique applied at the transmitter that maps the user data streams to spatially distributed antenna signals, based on CSI. The goal is to optimize transmission in terms of power efficiency, SINR, and user fairness. Classic linear precoding schemes include:

1. *ZF*: completely removes inter-user interference by setting $\mathbf{W}_{\mathrm{ZF}} = \mathbf{H}^{\mathrm{H}}\Big(\mathbf{H}\mathbf{H}^{\mathrm{H}}\Big)^{-1}$, at the

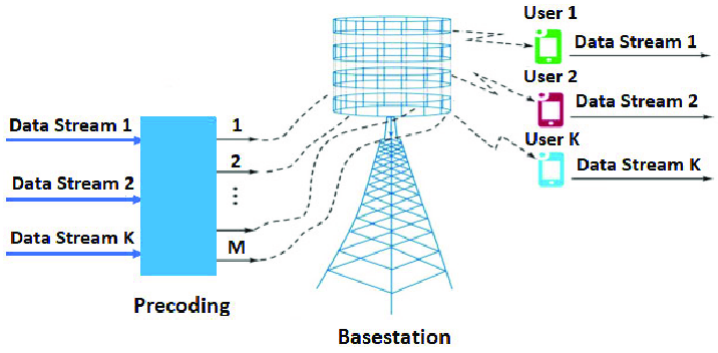


Figure 2.1 Downlink mMIMO System Model [2].

cost of possible noise amplification if $\mathbf{H}$ is ill-conditioned;

2. *Maximum-Ratio Transmission (MRT)*: maximises the received power in each user direction by setting $\mathbf{w}_k \propto \mathbf{h}_k^{\mathrm{H}}$, thereby enhancing array gain but leaving residual inter-user interference untouched;

3. *WMMSE*: iteratively minimizes a weighted mean-square error or equivalently maximizes a weighted Sum-Rate (SR) yielding near-optimal SINR performance at the expense of higher online complexity due to its alternating-optimization updates [16].

Non-linear schemes [17] offer still higher spectral efficiency but demand prohibitively complex hardware for large-scale arrays, motivating the recent turn toward data-driven, DL alternatives reviewed in the sequel.

ZF excels at canceling inter-user interference but suffers from noise-boosting whenever the channel Gram matrix $\mathbf{H}\mathbf{H}^{\mathrm{H}}$ is ill-conditioned [18], whereas MRT is power-efficient yet oblivious to cross-talk interference; WMMSE closes most of the performance gap to the capacity limit, though at a heavy computational price because its alternating updates must run every coherence block. In the canonical *multi-user* mMIMO setting, a BS with M antennas serves K single-antenna terminals simultaneously ($M \gg K$ is typical). The composite downlink channel is

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_1^{\mathrm{H}} \\ \vdots \\ \mathbf{h}_K^{\mathrm{H}} \end{bmatrix} \in \mathbb{C}^{K \times M}, \tag{2.3}$$

and the precoder must manage the mutual interference among these K user channels. By contrast, *single-user* (point-to-point) MIMO involves just one terminal equipped with multiple antennas; there multi-user interference disappears and eigen-beamforming suffices.

A widely studied design objective for the MU case is to minimize the total transmit power while guaranteeing a target SINR$=\Gamma_k$ for every user:

$$\begin{aligned} \underset{\{\mathbf{w}_k\}_{k=1}^{K}}{\text{minimize}} \quad & \sum_{k=1}^{K} \left\| \mathbf{w}_k \right\|_2^2 \\ \text{subject to} \quad & \frac{\left| \mathbf{h}_k^{\mathrm{H}} \mathbf{w}_k \right|^2}{\sum_{j \neq k} \left| \mathbf{h}_k^{\mathrm{H}} \mathbf{w}_j \right|^2 + \sigma_n^2} \;\geq\; \Gamma_k, \quad k = 1, \ldots, K. \end{aligned} \tag{2.4}$$

Problem (2.4) can be cast as a second-order cone program and solved by convex optimization, or approximated efficiently by the iterative WMMSE algorithm [16] highlighting the classical trade-off between optimality and online complexity that motivates the DL approaches, which are reviewed next.

Beyond their celebrated SE gains, mMIMO arrays unlock a host of additional advantages across diverse wireless scenarios. In Millimeter Wave (mm-Wave) bands, where free-space path loss is severe, the pencil beams afforded by hundreds of antennas provide the array gain required to sustain gigabit links. For the Internet-Of-Things (IOT), an M-element array can spatially multiplex thousands of ultra-low-power devices while maintaining sub-milliwatt per-terminal transmit power, thereby pushing network–wide EE to new heights [19]. In ultra-reliable low-latency communications (URLLC) [20] and edge-computing slices, the same spatial degrees of freedom translate into diversity gains and the ability to serve many latency-sensitive tasks concurrently. Large arrays are equally attractive for wireless back-haul: narrowly steered beams form high-capacity, interference-limited links between BS without expensive fibre deployment.

Crucially, all of these benefits hinge on accurate CSI at the transmitter, which is typically acquired via uplink training and channel reciprocity in time-division duplex (TDD) systems, so that the precoding/beamforming vectors can be matched to the instantaneous propagation environment. While advanced linear precoders such as ZF and WMMSE can, in principle, deliver near-optimal performance. Their matrix inversions or iterative updates scale poorly with the antenna count M and user count K. This computational burden has sparked interest in data-driven alternatives.

Recent work shows that DNNs can learn a direct mapping from CSI to high-quality precoding matrices, thereby bypassing the need to solve an optimization problem for every new channel realization. For example, Elbir and Papazafeiropoulos proposed a deep CNN–based hybrid precoder for multi-user mm-Wave mMIMO that outperforms conventional optimization while cutting run-time by orders of magnitude [21]. Huang *et al.* demonstrated a related

approach that matches, and sometimes exceeds, the SE of classic algorithms at a fraction of their latency [22]. These studies confirm that neural networks can approximate optimal beamforming strategies, adapt on the fly to the channel fluctuations, and exhibit robustness to CSI imperfections when such impairments are included during training.

Motivated by these successes, this thesis adopts the premise that DL can enable real-time precoding even for very large arrays. We extend the DL-based paradigm by tackling two practical obstacles: EE and deployability on hardware-constrained BS accelerators. The remainder of this chapter reviews the necessary background in DL for wireless, and introduces the energy-consumption models relevant to mMIMO, and surveys model compression techniques, quantization, pruning, and NAS that will later be fused into our BitAdapt framework.

2.2 DNN in Wireless Communications

DL is a branch of Machine Learning (ML) that employs multi-layer Artificial Neural Networks (ANNs) to automatically learn representations of data. Each layer of a DNN performs a linear transformation (weighted sums) followed by a nonlinear activation, enabling the network to model complex, high-dimensional relationships. During training, typically via stochastic gradient descent and back-propagation, the network's millions of weight parameters are adjusted to minimize a loss function measuring the discrepancy between predictions and ground truth. Thanks to the composition of many nonlinear layers, DNNs can approximate extremely complex functions, which has led to breakthroughs in image recognition, Natural Language Processing (NLP), and game playing.

The success of DNNs in those domains has spurred their application to wireless communication problems, which often involve complicated, stochastic mappings. Unlike traditional signal processing algorithms that rely on hand-crafted models and assumptions (Gaussian noise, linear channels, convex optimization, *etc.*), a DL approach can, in principle, learn the system's behavior directly from data. This data-driven paradigm is attractive when the true model is highly complex or partially unknown.

In the wireless physical layer, researchers have demonstrated DNN-based solutions for channel estimation, detection, resource allocation, and precoding. For example, a neural network can learn to estimate CSI from pilot signals even when standard linear estimators falter due to non-idealities or sparse pilots, essentially learning to interpolate or extrapolate CSI in a manner that adapts to the propagation environment. In the context of mMIMO precoding, a DNN can be trained to map an input CSI matrix $\mathbf{H}$ (or its compressed representation) to an output precoding matrix $\mathbf{W}$ that maximizes a performance metric such as SR or fairness. This

bypasses solving an optimization problem anew for each $\mathbf{H}$; once trained, the DNN produces an approximate solution in a single forward pass. The key advantage is speed: inference can be executed in microseconds on specialized hardware, whereas solving a convex optimization (e.g., WMMSE) or inverting large matrices for ZF may be orders of magnitude slower as M, K grow. Such speed is crucial for real-time adaptation in fast-fading channels or low-latency applications. Moreover, DNNs may uncover analytically elusive precoding strategies, especially in non-linear or non-convex formulations. Recent studies confirm this shift in practice, with Transformer architectures increasingly applied to CSI feedback and channel estimation in mMIMO, while deep-unfolded WMMSE networks show that hybrid model-based/data-driven designs remain competitive [23, 24]. This aligns with survey evidence on the maturing role of deep learning in precoding and feedback [25].

Early explorations of DL in wireless have shown promising gains. Deep networks have been used to cancel interference and to perform hybrid analog or digital beamforming in mmWave systems, achieving performance close to ideal algorithms under challenging hardware constraints [26]. Convolutional and recurrent architectures exploit spatial and temporal channel correlations. A CNN can extract spatial features from an antenna array's CSI, whereas an Recurrent Neural Networks (RNN) or Long Short-Term Memory (LSTM) captures temporal evolution for prediction or tracking. A study by Chen *et al.* [27] applied DL for mm-Wave interference cancellation and demonstrated improved signal quality compared with traditional nulling techniques, particularly under hardware imperfections. Likewise, recent work shows that with sufficient training data across diverse scenarios, a single DNN can generalize across deployment conditions (e.g. different cell layouts or mobility patterns), whereas model-based schemes require re-derivation or re-calibration for each case [28]. Building on this trend, [29] introduced a Swin Transformer-based network that significantly improves CSI feedback efficiency in massive MIMO, while [30] proposed SemCSINet, a semantic-aware CSI compression method that embeds CQI information, thereby improving robustness under noisy and low-SNR conditions.

In parallel, model-inspired approaches combine optimization with deep architectures. Notably, [31] developed a deep graph unfolding framework for MU-MIMO beamforming that bridges classical WMMSE optimization with graph neural networks, achieving both efficiency and scalability. These advances highlight the rapid evolution of DNN-based methods from black-box estimators toward architectures that integrate domain knowledge while addressing robustness and efficiency.

Despite these advantages, deploying DNNs in wireless systems raises practical challenges. A typical deep model may contain millions of parameters and demand billions of arithmetic op-

erations per inference. Implementing such networks on power and cost-constrained hardware (BS processors, mobile devices, or edge units) is non-trivial. The computational load can overwhelm Digital Signal Processing (DSP) or Field-Programmable Gate Array (FPGA) resources if not optimized, and the memory footprint for weights and activations often exceeds on-chip capacity, forcing energy-expensive off-chip accesses. Furthermore, the black-box nature of DNNs means performance can degrade under conditions poorly represented in the training set, raising concerns about reliability and robustness in mission-critical links.

These issues motivate research into efficient DL for communications networks that are not only accurate but also compact, fast, and energy-efficient. Techniques such as network pruning, weight quantization, Knowledge Distillation (KD), and NAS are essential to bridge the gap between modern DNN capabilities and the strict real-time requirements of wireless systems. In this thesis, we focus on quantization and compact network design for a DL-based precoding system. By compressing the neural precoder model and tailoring its architecture, we aim to realize the benefits of DL (high SE and adaptability) within the power and hardware constraints of practical mMIMO BSs or edge devices.

2.3 Energy Efficiency in Massive MIMO Systems

In evaluating any new wireless processing technique (such as a DL-based precoding algorithm), it is crucial to consider its energy profile. We distinguish between two related metrics: the absolute energy consumption of the system and its EE, defined as the useful throughput achieved per unit of energy consumed. Next, we define these terms more formally and discuss their significance in mMIMO systems.

Energy Consumption. Energy consumption refers to the total electrical energy required for the system to perform its functions, for example, computing a precoding matrix or running a neural network inference, typically measured in joules per operation or per time interval. In a digital system, energy consumption has two primary components:

1. *Computation energy* (E_{comp}): the energy spent on arithmetic and logical operations (such as the Multiply-Accumulate (MAC) operations in matrix multiplications).

2. *Data-movement energy* (E_{data}): the energy used to transport data between different parts of the system, especially across memory hierarchies (registers, caches, main memory) and processing units.

In modern processors, moving data can be far more energy-costly than performing arithmetic.

We can express the total energy for an operation as

$$E_{\text{total}} = E_{\text{comp}} + E_{\text{data}} \,, \tag{2.5}$$

with

$$E_{\text{data}} = \sum_{\ell} A_{\ell} \, E_{\text{access},\ell} \,, \tag{2.6}$$

and where A_ℓ is the number of accesses to a given memory level ℓ, and $E_{\text{access},\ell}$ is the energy per access at that level.

On-chip versus off-chip memory. Accessing on-chip Static Random Access Memory (SRAM) or cache typically costs only a few PicoJoule (pJ) per 32-bit word, whereas an off-chip Dynamic Random Access Memory (DRAM) access can cost two orders of magnitude more (500–1000 pJ per 32-bit word) [32]. Results from an empirical study are shown in Table 2.1.

Table 2.1 Typical energy cost of a 32-bit operation or memory access in a 45 nm CMOS process, adapted from Horowitz [1].

Operation / Access	Energy [pJ]	Relative cost
32-bit MAC	3–5	1×
32-bit SRAM	5–20	$\sim 3\times$
32-bit DRAM	∼800	$\sim 200\times$

Consequently, data movement often dominates the energy budget: in many DNN or baseband workloads, 75–90% of the total energy is attributed to memory traffic [33].

Energy Efficiency. In wireless communications, EE is usually defined as the ratio of achieved throughput to the energy consumed. For an mMIMO system, a common EE metric is the sum SE per Joule of energy, measured in (bits/s)/W or, equivalently, bits/Joule. If R_{sum} denotes the sum rate (in bits/s) of all users and P_{cons} denotes the total power consumption of the system, the energy efficiency is given by

$$\text{EE} \;=\; \frac{R_{\text{sum}}}{P_{\text{cons}}} \quad \left[\text{bits J}^{-1}\right]. \tag{2.7}$$

Higher EE means the system delivers more throughput for each joule of energy expended. Importantly, maximizing EE is not the same as minimizing energy consumption or maximizing throughput alone; it is about finding an optimal trade-off. A system operating at

extremely low power might achieve low throughput and thus only mediocre EE, while one with huge throughput but astronomical power usage could also exhibit poor EE. The sweet spot is often at an intermediate operating point.

These insights collectively highlight a converging theme in the field: achieving high EE in DL-based mMIMO systems requires a joint consideration of both algorithmic and hardware aspects. In the context of this thesis, the proposed DL-based precoding framework adheres to these principles by prioritizing on-chip data reuse, employing model compression techniques such as pruning and quantization to ensure that neural network weights fit within SRAM, and embracing algorithm–hardware co-design methodologies. These design strategies are fundamental to ensuring that the developed solutions meet the stringent energy constraints of practical wireless communication systems. In the subsequent sections, we delve deeper into two specific techniques: neural network quantization and NAS, which are pivotal in shaping DNNs capable of delivering high performance while operating within realistic energy budgets for mMIMO precoding applications.

mMIMO systems have revolutionized modern wireless communications by delivering substantial spectral and spatial multiplexing gains through large-scale antenna arrays. However, the significant infrastructure required—comprising hundreds of Radio Frequency (RF) chains, power amplifiers, and baseband processing units—translates into substantial energy consumption. The result is a pressing need for EE, and critical for both BS operation and edge-device deployment.

Early analytical models by Björnson *et al.* [34, 35] decomposed system power into static (circuit and cooling) and dynamic (transmission and baseband) components. Their rigorous analysis of ZF precoding revealed that EE peaks in intermediate operational regimes, even advocating increased per-antenna transmit power as antenna count grows. These studies decisively demonstrated that maximum SE does not equate to maximum EE.

Meanwhile, hardware-level strategies such as antenna selection [36] and hybrid analog-digital architectures [37] targeted static power reduction. By disabling underutilized RF chains or leveraging analog beamforming to reduce digital hardware, researchers achieved meaningful EE improvements while incurring minimal performance penalties. Yet these methods often restrict beamforming flexibility, hindering performance in dynamic propagation environments.

On the algorithmic side, processing energy became a limiting factor. Although ZF and regularized ZF remain popular for their ability to suppress interference efficiently, their computational cost scales poorly with antenna count due to matrix inversions [38]. More complex methods like WMMSE enhance throughput but impose steep energy and latency costs, an

unsuitable trade-off for real-time mMIMO systems.

Quantization and low-resolution hardware have since been explored to temper dynamic power demands. For instance, reduced-bit Dynamic Algorithm Configurations (DACs) and quantized feedback schemes can substantially decelerate power consumption. However, increased quantization noise introduces performance degradation, complicating deployment unless carefully compensated.

With the advent of DL, data-driven precoders emerged to reduce computation time and energy cost by approximating optimal beamforming maps. CNN and autoencoder-based precoders such as those proposed by Elbir and Papazafeiropoulos [21] demonstrated near WMMSE performance with reduced inference complexity. While promising, these models typically require significant on-chip memory for parameters and intermediate activations, inflating energy consumption, especially when transfers to off-chip DRAM occur.

In response, techniques from the DNN compression literature, such as pruning, quantization, and architecture search, have been applied, producing lightweight DL-based precoders. Nagel et al. [39] showed that aggressively quantized models maintain high performance while reducing computational energy. Similarly, hybrid DL–analytical frameworks incorporate domain knowledge (e.g., ZF structure embedded within a neural net), achieving performance gains with reduced parameter counts.

A critical realization from DNN accelerator research underscores the importance of memory access: off-chip DRAM transfers are an order of magnitude more energy-intensive than on-chip SRAM access, which in turn is significantly more expensive than pure MAC operations. In seminal work, Han *et al.* introduced the EIE accelerator for compressed DNNs, demonstrating that fitting a model entirely in SRAM leads to $120\times$ energy savings over DRAM-based designs by exploiting weight sparsity and quantization [40]. EIE's architecture shows the profound impact of minimizing off-chip data movement, a principle directly applicable to DL-powered mMIMO precoders.

Further innovations such as EDEN [41] push this concept by leveraging approximate DRAM with lower voltage operation. Paired with retraining techniques to compensate for reduced precision, EDEN achieves up to 37% DRAM energy reductions with minimal accuracy loss, offering a new dimension for energy-efficient inference.

Together, these works underscore a converging trend: energy-efficient mMIMO design demands a holistic treatment of both computation and memory. While prior methods focusing on hardware or signal processing deliver isolated benefits, the integration of DL with careful memory optimization, particularly strategies to confine model inference within on-chip mem-

ory, emerges as a powerful paradigm. The field is gravitating toward systems that minimize costly memory transfers, employ model compression techniques, and align signal processing with hardware-aware design, all to achieve optimal EE in real-world deployments.

2.4 Quantization Methods for Neural Networks

As introduced earlier, quantization is a model compression technique that reduces the precision of numbers used to represent neural network parameters and computations. The goal of quantization is to replace Floating-Point Single Precision (FP32) with lower-bit integer (or fixed-point) operations, thereby reducing memory usage, speeding up computation (especially on hardware like DSPs or integer-arithmetic accelerators), and most pertinently, lowering energy consumption. Quantization can be applied to network weights, activations, gradients, or all of the above.

2.4.1 Quantization Fundamentals

Let $\mathcal{R} \subset \mathbb{R}$ be the original set of values to be quantized, and $r \in \mathcal{R}$. Quantization maps r to a lower-precision value $\hat{r} \in \hat{\mathcal{R}} \subset \mathbb{Z}$ using a set of quantization parameters including a scaling factor S, a zero-point offset Z, and a clipping range $[\alpha, \beta]$. The general form of uniform quantization is:

$$\hat{r} = Q_u(r, S, Z, B) = \left\lfloor \frac{\text{clip}(r, \alpha, \beta)}{S} - Z \right\rceil , \tag{2.8}$$

where the scale and zero-point are given by:

$$S = \frac{\beta - \alpha}{2^B - 1}, \quad Z = \frac{-\alpha}{S} . \tag{2.9}$$

The corresponding dequantization function is:

$$\tilde{r} = D_u(\hat{r}, S, Z) = (\hat{r} + Z) \cdot S . \tag{2.10}$$

The quantization error $e_q = \tilde{r} - r$ arises from rounding and clipping.

2.4.2 Uniform and Non-Uniform Quantization

The clipping bounds α and β are typically chosen to cover a high percentage of the values of r (e.g., $\alpha = \min(r)$, $\beta = \max(r)$, or some percentile-based range). Any r outside $[\alpha, \beta]$ will be saturated to the nearest bound, introducing clipping error. Reducing this range can improve the effective resolution for in-range values at the cost of more frequent clipping of outliers.

If the distribution of r is roughly uniform or not too heavy-tailed, uniform quantization works well. However, if r has a highly skewed distribution (for example, activations in ReLU networks can have many small values and a few large ones), non-uniform quantization may be beneficial. One common non-uniform scheme is a logarithmic or power-of-two quantization where levels are spaced geometrically. For instance, a simple base-2 logarithmic quantizer might map a real r to

$$\hat{r} = Q_{nu}(r, B) = \text{clip}\left(\lfloor \log_2(|r|) \rceil, 0, 2^{B-1}\right), \tag{2.11}$$

followed by the corresponding dequantization:

$$\tilde{r} = \begin{cases} 0 & \text{if } \hat{r} = 0 \\ 2^{\hat{r}} & \text{otherwise.} \end{cases} \tag{2.12}$$

2.4.3 Clipping and Rounding

Clipping narrows the input domain $[\min(\mathcal{R}), \max(\mathcal{R})]$ to a target range $[\alpha, \beta]$, which is then quantized. It reduces rounding errors but can introduce clipping error for outlier values. The clip function is defined as

$$\text{clip}(r, \alpha, \beta) = \begin{cases} \alpha & \text{if } r < \alpha\,, \\ r & \text{if } \alpha \leq r \leq \beta\,, \\ \beta & \text{if } r > \beta\,, \end{cases}$$

ensuring r lies in $[\alpha, \beta]$. After clipping, rounding is applied to map the scaled value $(r/S - Z)$ to the nearest integer (which becomes $\hat{r}$). While standard "round-to-nearest" is used widely, advanced strategies exist to minimize the impact of rounding on model accuracy. One such approach is Hessian-aware rounding [39], which uses second-order information (the Hessian of the loss function L concerning weights) to decide how to round each weight. Intuitively, if a weight is in a "flat" region of the loss surface, a larger rounding error can be tolerated; but if the loss is highly sensitive to that weight (steep curvature), rounding it might cause a big drop in accuracy. The Hessian-aware scheme attempts to minimize the expected loss increase due to rounding

$$\mathbb{E}[L(w + \Delta w) - L(w)] \approx \frac{1}{2} \Delta w^T H(w) \Delta w \tag{2.13}$$

and chooses rounding offsets Δw that keep this quantity small. In practice, such methods may be implemented by formulating rounding as an optimization problem over a small set of choices. Another consideration is whether to use symmetric quantization (where zero in real domain maps to zero in quantized domain, i.e. $\alpha = -\beta$ and $Z = 0$) or asymmetric quantization (allowing $\alpha \neq -\beta$ and a non-zero Z). Symmetric quantization is simpler and often used for network weights, which have symmetric distributions around 0. Asymmetric quantization is common for activations which are non-negative (like Rectified Linear Unit (ReLU) outputs): by setting Z such that it represents the real value 0, we effectively use the full range of quantized values to cover $[0, \beta]$ rather than wasting half the codes on negative values that activations never take. The granularity at which quantization parameters (S, Z) are chosen also matters. We can quantize an entire tensor with one scale (layer-wise quantization), or use a separate scale per output channel, per group of channels, or even per weight (fine-grained channel-wise or group-wise quantization). Finer granularity can improve accuracy because it adapts to local statistics of weights/activations (e.g., each convolutional filter may have different variance and range, so giving each its own S is beneficial). However, it also increases the overhead of storing scale factors and might complicate efficient hardware implementation. To evaluate quantization effects without deploying actual low-precision hardware, one often uses quantization simulation (sometimes called fake quantization). During simulation, each operation in the computational graph is emulated in FP32 but with inserted quantize-dequantize steps to mimic the finite precision. For example, one can take an activation tensor x in a forward pass and compute

$$\tilde{r} = G_u(r, S, Z, B) = D_u(Q_u(r, S, Z, B), S, Z)\,. \tag{2.14}$$

This technique helps assess quantization impact on accuracy without deploying the quantized model, and enables selection of optimal per-layer quantization parameters.

Post-Training Quantization (PTQ) PTQ is the most straightforward quantization approach, as it simply quantizes a pre-trained model's parameters after training, with no additional model retraining. In PTQ, the weights (and usually the activations) of a neural network are statically converted from FP32 to a lower-bit fixed-point format (such as 8-bit integers). The appeal of PTQ lies in its simplicity; it's fast to implement and doesn't require any further training data or compute cycles. However, the accuracy of a model can drop sharply under aggressive PTQ, say when using only 2–4 bits for all layers. Uniformly applying a very low precision to every layer tends to hurt the most sensitive layers (typically those like the first convolution or final classification layer), resulting in notable performance degra-

dation in many cases [42]. Recent techniques (e.g., bias correction or weight equalization [43]) can mitigate some of these issues, allowing slightly better accuracy without retraining. Even so, PTQ is generally best suited for models that are naturally robust to quantization or in scenarios where a modest loss in accuracy is acceptable.

2.4.4 Quantization-Aware Training (QAT)

QAT embeds quantization operations directly into the training pipeline. During each forward pass, fake quantization simulates low-bit arithmetic. Backpropagation uses the Straight Through Estimator (STE)) to approximate the gradients:

$$\frac{\partial q(x)}{\partial x} \approx 1$$

Quantizers are attached to weights, biases, and activations. QAT requires labeled data and multiple epochs of fine-tuning. While computationally demanding, it offers superior accuracy retention, especially at extreme quantization levels (e.g., INT4, INT2).

Learned Step Size Quantization (LSQ). LSQ can be seen as an extension of QAT where the quantizer itself has trainable parameters. In LSQ, the step size (the scaling factor that determines the quantization levels) is not fixed ahead of time; instead, its value is learned via backpropagation along with the network weights [44]. Practically, LSQ inserts a small number of new parameters representing step sizes for different layers or tensors, and uses the STE technique to approximate gradients through the rounding operation (which is non-differentiable). By learning optimal step sizes, LSQ often achieves better accuracy at extremely low bit-widths because it can finely adjust the quantization resolution where needed. This approach is particularly useful when it's unclear how to manually choose the best quantization intervals, for instance, when balancing accuracy vs. EE on different hardware platforms. LSQ has proven effective and has been adopted in both research contexts and in hardware-aware neural network design.

2.4.5 Mixed-Precision Quantization (MPQ)

MPQ assigns different bit-widths to different layers, channels, or tensor components. This reflects that not all parts of a network are equally sensitive to quantization. Sensitive layers use higher precision (e.g., 8 bits), while robust ones are quantized more aggressively (e.g., 2 or 4 bits).

Let each layer l use bit-width B_l, then

$$\hat{r}_l = Q(r_l, S_l, Z_l, B_l)\,.$$

Finding optimal assignments is a combinatorial challenge. Search strategies include:

- Exhaustive search (feasible for small networks),
- Gradient-based methods like Differentiable Quantization (DQ) [45],
- Sensitivity-based methods like HAWQ [46],
- Reinforcement Learning (RL) approaches.

MPQ can also be applied channel-wise or kernel-wise. While it provides better trade-offs, the benefits are hardware-dependent, requiring hardware-aware search and deployment strategies.

2.4.6 BitPruning Quantization

Recent advances in neural network compression have increasingly focused on bit-level pruning strategies to enable energy-efficient deployment of deep learning models in resource-constrained environments. Traditional approaches, such as weight pruning and structured pruning, aimed at removing individual weights or entire filters to reduce model size and computation. While effective, these methods often lacked the fine granularity required for real hardware efficiency. To address this, researchers began exploring bit-level optimization, where the precision of each weight or activation is reduced based on its relative contribution to model performance. This shift gave rise to quantization-aware pruning techniques that jointly learn parameter values and their optimal bitwidths during training.

Among these, BitPruning [12] emerged as a pivotal approach by treating bitwidths as differentiable, learnable variables. This allows each layer or even individual parameters to adopt a precision level that balances accuracy and efficiency. Extensions of BitPruning have further enhanced its hardware compatibility. For instance, Sekikawa and Yashima [47] reformulated dot products into bit-shift and addition operations, eliminating multiplications and reducing energy cost. Parallel works such as LQ-Nets [48] and PACT [49] focused on learning quantization levels and clipping thresholds, while Soft Filter Pruning [50] maintained differentiability in pruning masks to allow gradient-based updates. Moreover, hardware-aware strategies have gained traction, including RL-based frameworks [51], ultra-low precision arithmetic [52], and memory-efficient architectures like BitS-Net [53]. Collectively, these efforts highlight a growing trend: the convergence of pruning and quantization into unified, learnable frameworks

that offer both compression and EE, paving the way for deployable Artificial Intelligence (AI) in communication systems, edge devices, and emerging memory technologies.

Summary and Open Challenges

Despite substantial progress, several limitations remain. Uniform-precision schemes ignore the heterogeneous sensitivity of layers; MPQ improves flexibility but suffers from a combinatorial search space; LSQ learns step sizes yet typically assumes fixed per-layer bit widths; and BitPruning offers fine-grained control at the cost of training and deployment complexity. Moreover, many methods abstract away deployment constraints, energy and memory costs, latency/throughput targets, and, critically, the availability of hardware support for variable precision, which are central in mMIMO precoding.

These gaps motivate frameworks that are not only precision-aware but also hardware-, deployment-, and energy-aware. Accordingly, the next subsection (Section 2.4.7) examines hardware support for variable bit-width, surveying accelerator and FPGA platforms that enable per-layer mixed-precision execution and clarifying the feasibility assumptions underpinning our energy-aware regularization.

2.4.7 Hardware Support for Variable Bit-Width

The practicality of mixed-precision quantization ultimately hinges on the capabilities of the target hardware. While commodity GPUs and DSPs typically operate at a fixed arithmetic width (e.g., FP16 or INT8), two classes of platforms increasingly support per-layer precision variation at compile time or even runtime.

The first type are the reconfigurable processing elements (PEs) that can be found in custom DL accelerators. These designs allow the multiply–accumulate (MAC) datapath to be spatially partitioned, enabling dynamic subword parallelism. A single PE can, for example, issue one INT16 MAC per cycle, or equivalently two INT8 or four INT4 MACs in parallel, depending on the layer's assigned precision [54, 55]. Such flexible-width arrays map naturally to layer-heterogeneous bit-width schedules required by mixed-precision quantization.

A second implementation approach supporting mixed precision models is an FPGA-based pipelined implementation. Here, each network layer is realized as a dedicated streaming stage whose arithmetic precision is tailored to that layer's sensitivity (e.g., INT8 in early layers, INT4 in mid blocks, INT2 in late layers). Although this specialization can be resource-intensive, heavy quantization substantially reduces logic and DSP utilization, making per-layer precision practical in many cases [**?**, 56]. Pipeline specialization has been demonstrated

across quantized CNN/RNN accelerators, preserving accuracy while improving energy efficiency.

In summary, while per-layer adjustable precision is not yet universal, reconfigurable MAC arrays in custom ASICs and multi-precision pipelines on FPGAs make it increasingly realistic. Consequently, evaluations of mixed precision (and related methods such as BitPruning) should differentiate algorithmic potential from deployment feasibility and consider platforms that explicitly expose variable bit-width execution.

2.5 Neural Architecture Search (NAS)

NAS is an emerging paradigm in DL that aims to automate the design of neural network architectures by optimizing their structure for a specific task or hardware constraint. NAS techniques are particularly valuable for applications requiring custom performance trade-offs, such as edge inference, where accuracy, latency, and energy efficiency must be carefully balanced. In the context of wireless communication systems such as mMIMO precoding, NAS offers the potential to discover architectures that are well-adapted to deployment-specific requirements.

Early NAS approaches relied heavily on RL or evolutionary algorithms to explore architecture spaces [57]. These methods produced high-performing models but were prohibitively expensive computationally, often requiring thousands of GPU-days. To reduce the search cost, differentiable NAS (DARTS) was introduced [45], which relaxes the discrete architecture search into a continuous optimization problem using gradient descent. While DARTS significantly accelerates search time, it often suffers from performance collapse due to instability during the architecture weight optimization process.

More recent NAS frameworks have shifted toward hardware-aware NAS [58], where search objectives explicitly account for resource constraints such as latency, memory access, or energy consumption. These methods optimize a multi-objective loss function involving both accuracy and hardware cost models. ProxylessNAS [59] further improves this idea by eliminating the need for proxy tasks and directly searching for efficient architectures on the target device. Despite their effectiveness, these approaches rely on accurate profiling of the hardware and can be sensitive to variations in the deployment platform.

NAS has also been extended to support quantization-aware and mixed-precision training. Recent works jointly search over both architectural and quantization design spaces [46]. These hybrid methods discover models that are optimized in both structure and numerical precision, significantly improving energy efficiency. However, such joint searches increase

the dimensionality of the search space, making the optimization problem more complex and sensitive to initialization and regularization.

In wireless communication applications, NAS remains relatively underexplored. A few studies have proposed customized search spaces tailored to physical-layer tasks sensing and modulation recognition [60]. These domain-specific NAS approaches reduce search time by constraining architectural choices and leveraging knowledge about signal processing pipelines. Nonetheless, most existing NAS methods are designed for general-purpose vision or language tasks, and their adaptation to communication systems requires careful redesign of search objectives, datasets, and performance metrics.

Overall, NAS presents a promising tool for designing efficient DL models tailored to specific hardware and system constraints. However, challenges remain in improving the scalability of search algorithms, integrating energy and latency models accurately, and adapting NAS frameworks to domain-specific use cases such as wireless communication and mMIMO systems.

2.6 Site-Specific Optimization of Deep Learning Models

In real-world wireless communication systems, the performance of DL-based models, particularly for tasks such as channel estimation, beamforming, and precoding, can vary substantially across different deployment environments. Factors such as user mobility, building geometry, material reflections, and signal blockage result in site-dependent propagation characteristics that affect CSI. Despite these known variations, most existing DL models are trained under generalized or averaged conditions, limiting their adaptability and robustness in site-specific scenarios. Throughout this thesis, "site-specific" refers to the *training/evaluation setup*—we train a separate model for each deployment scene rather than a restriction of the compression method itself, which is general and site-agnostic. Traditional approaches to site-specific optimization involve retraining models for each environment using locally collected data [61].

In site-specific optimization for wireless PHY tasks (channel estimation, beamforming, precoding), performance can vary markedly across deployments because propagation depends on user mobility, geometry, materials, and blockage, which shape the observed CSI. A common strategy in the literature is to retrain models with locally collected data for each environment [61]. While effective, per-site retraining introduces costs in data acquisition, curation, and lifecycle management, and can increase computational and memory overhead when many specialized models must be maintained. Recent studies therefore analyze the accuracy efficiency trade-off between site-specific tailoring and model sharing, including generalization

across heterogeneous deployment conditions.

To study these trade-offs under controlled, reproducible settings, many works employ ray-tracing–generated datasets as benchmarks to emulate site-specific conditions (e.g., DeepMIMO [62]). Such synthetic datasets capture spatial channel structure and geometry and are well-suited for supervised learning and ablations; however, they serve as proxies rather than substitutes for field measurements and may not reflect all hardware or traffic non-idealities. As a middle ground between per-site retraining and fully shared models, the literature also explores lightweight adaptation, where a pre-trained model is fine-tuned with a small number of target-site samples [63], as well as meta-/few-shot approaches aimed at improving portability across diverse environments [64].

For instance, the DeepMIMO dataset [62] provides a flexible framework for generating spatially-aware channel data using real-world 3D maps and ray-tracing engines. These synthetic datasets allow DL models to learn location-dependent CSI patterns and spatial channel correlations.

An alternative to full retraining is model adaptation, where a pre-trained generic model is fine-tuned or adapted using a small number of samples from the target site [63]. Meta-learning and few-shot learning techniques have been explored in this context to improve model generalization across diverse environments [64]. These methods reduce the need for large-scale retraining while maintaining high performance in new locations. However, their effectiveness depends on the similarity between training and deployment environments, and the availability of representative samples from the new site.

Some recent works have incorporated site-specific optimization into architectural design via model compression and quantization. These studies exploit spatial channel sparsity or hardware heterogeneity present in different deployment scenarios [65]. Techniques like pruning, QAT, and NAS have been employed to discover models that are not only energy-efficient but also site-tuned, either by adapting model capacity to site conditions or by embedding location-specific channel features into the model itself.

Despite these advances, challenges remain in efficiently adapting DL models to site-specific characteristics without incurring excessive overhead. Existing methods typically rely on full or partial retraining, which may not scale well in distributed or low-latency applications. Furthermore, the integration of site-specific constraints with other objectives such as energy efficiency, latency, or hardware awareness is still underexplored, especially in the context of real-time systems like mMIMO precoding.

These limitations highlight the need for DL frameworks that can generalize across sites while

offering mechanisms for lightweight, scalable adaptation. Future research in this area will likely focus on combining environmental modeling with compression-aware learning strategies to improve the practicality and efficiency of site-specific DL deployments in wireless systems.

CHAPTER 3 ARTICLE 1: COMPRESSION OF SITE-SPECIFIC DEEP NEURAL NETWORKS FOR MASSIVE MIMO PRECODING

Ghazal Kasalaee, Ali Hasanzadeh Karkan, Jean-François Frigon, and François Leduc-Primeau

Published in: IEEE International Conference on Machine Learning for Communication and Networking

Publication date: February 8, 2025

3.1 Abstract

The deployment of deep learning (DL) models for precoding in massive multiple-input multiple-output (mMIMO) systems is often constrained by high computational demands and energy consumption. In this paper, we investigate the compute energy efficiency of mMIMO precoders using DL-based approaches, comparing them to conventional methods such as zero forcing and weighted minimum mean square error (WMMSE). Our energy consumption model accounts for both memory access and calculation energy within DL accelerators. We propose a framework that incorporates mixed-precision quantization-aware training and neural architecture search to reduce energy usage without compromising accuracy. Using a ray-tracing dataset covering various base station sites, we analyze how site-specific conditions affect the energy efficiency of compressed models. Our results show that deep neural network compression generates precoders with up to 35 times higher energy efficiency than WMMSE at equal performance, depending on the scenario and the desired rate. These results establish a foundation and a benchmark for the development of energy-efficient DL-based mMIMO precoders.

3.2 Introduction

The deployment of DL models in mMIMO systems has shown great potential to advance wireless communication, particularly to optimize beamforming strategies. DNNs can dynamically adjust the beamforming to achieve better performance than traditional methods. However, the large size and overparameterization of these networks present challenges for practical deployment, especially on resource-constrained edge devices where memory, energy consumption, and processing latency are critically limited.

Designing energy-efficient communication systems is critical for the realization of mMIMO systems. However, finding a precoding solution that maximizes energy efficiency under low-resolution quantizer constraints remains a significant challenge [66]. Quantized CNNs are a popular choice for reducing hardware complexity. In CNNs, quantization can be applied to inputs, weights, and activations [67]. Typically, weights and activations are represented using 32-bit or 64-bit floating-point formats, but they can be quantized to lower-precision fixed-point representations, such as 2-bit or even 1-bit precision, to further reduce computational and memory requirements. Lowering the precision leads to a smaller model size; however, it can also reduce accuracy, so choosing the number of precision bits is a non-trivial task.

These constraints hinder the direct application of DNNs in real-world mMIMO systems. The celebrated WMMSE algorithm proposed in [7] is based on the equivalence between the SINR and mean square error, which is then solved using the block coordinate descent method. The WMMSE algorithm provides state-of-the-art performance and is widely used as a benchmark in the literature. However, existing iterative algorithms like WMMSE face challenges due to their computational complexity, which grows rapidly with the number of antennas and the network size. This increase in complexity leads to higher latency, making such algorithms less suitable for real-time applications or large-scale systems where rapid decision-making is critical.

To address these resource limitations, quantization techniques have emerged as an effective solution. Quantization reduces the complexity of neural networks by lowering the bit-width of model weights and activations, which decreases memory usage and energy consumption. Common approaches such as PTQ and QAT apply uniform precision across the entire network to optimize performance on resource-constrained platforms [68, 69]. However, these methods often fail to account for the varying sensitivity of different network layers to quantization, leading to inefficiencies and potential performance loss when aggressive quantization is applied uniformly.

In contrast, MPQ provides a more flexible approach by allowing different layers to use varying bit-widths, enabling a better balance between accuracy and energy efficiency [70]. By leveraging search algorithms such as Exhaustive Search, mixed-precision quantization can optimize bit-width allocation for specific layers, leading to significant reductions in energy consumption while maintaining high performance. This approach is particularly advantageous for complex models like DNNs, which are widely used in energy-efficient tasks such as real-time signal processing in mMIMO systems.

This paper proposes a mixed-precision quantization framework, augmented with Neural Architecture Search (NAS) [71], to optimize energy consumption in DL models used for mMIMO

beamforming. Our approach employs a streamlined neural architecture with a single CNN layer followed by three fully connected layers, designed to balance computational complexity and SINR performance. Additionally, by adapting our quantization strategy to site-specific conditions using ray-tracing datasets, we further enhance the model's energy efficiency without sacrificing accuracy or SINR performance.

In this paper, we address the challenge of enhancing energy efficiency in DL-based beamforming for mMIMO systems. We propose a mixed-precision quantization-aware framework that significantly reduces energy consumption compared to standard deep learning methods while maintaining competitive sum rate performance. This approach achieves results comparable to traditional techniques such as ZF and WMMSE, with a strong emphasis on energy efficiency. Additionally, we introduce site-specific model compression, a scalable method designed to adapt to unique site conditions. By leveraging site-specific ray-tracing datasets, this approach enables adaptive quantization that accounts for environmental factors, enhancing energy efficiency without compromising accuracy. This work establishes a solid foundation for advancing energy-efficient deep learning-based beamforming, offering a compelling alternative to conventional methods like ZF and WMMSE, particularly in scenarios where energy consumption is a critical concern.

3.3 System model

This work considers a multi-user mMIMO system where a BS with N_{T} transmit antennas serves N_{U} single-antenna users simultaneously. Let x_u represent the transmitted symbol for each user. The received signal at the u^{th} user can be represented as

$$\mathbf{y}_u = \mathbf{h}_u^{\dagger} \sum_{\forall u} \mathbf{w}_u x_u + \boldsymbol{\eta}\,, \tag{3.1}$$

in which $\mathbf{h}_u \in \mathbb{C}^{N_{\mathsf{T}} \times 1}$ is the wireless channel vector between the BS and u^{th} user; the term $\boldsymbol{\eta} \sim \mathcal{CN}(0, \sigma^2)$ denotes complex symmetric Gaussian noise with zero mean and a variance of σ^2; the downlink transmit Fully Digital Precoder (FDP) vector is denoted as $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_u, \ldots, \mathbf{w}_{N_{\mathsf{U}}}] \in \mathbb{C}^{N_{\mathsf{T}} \times N_{\mathsf{U}}}$. The corresponding SINR for user u is formulated as

$$\mathrm{SINR}(\mathbf{w}_u) = \frac{\left|\mathbf{h}_u^{\dagger}\mathbf{w}_u\right|^2}{\sum_{j \neq u} \left|\mathbf{h}_u^{\dagger}\mathbf{w}_j\right|^2 + \sigma^2}\,. \tag{3.2}$$

The goal is to find such a precoding $\mathbf{W}$ that maximizes the throughput under the maximal transmit power $P_{\max}$ constraint. Thus, the downlink sum rate maximization problem can be

formulated as

$$\max_{\mathbf{W}} R(\mathbf{W}), \tag{3.3}$$

$$\text{s.t.} \sum_{\forall u} \mathbf{w}_u^{\dagger} \mathbf{w}_u \leq P_{\mathsf{max}}, \tag{3.4}$$

where for a precoding matrix $\mathbf{W}$, the sum rate is

$$R(\mathbf{W}) = \sum_{\forall u} \log_2\Big(1 + \text{SINR}(\mathbf{w}_u)\Big). \tag{3.5}$$

3.3.1 Problem Definition

Beamforming is crucial in wireless communication, particularly at mmWave frequencies, where directional signal transmission is essential due to the limitations of omnidirectional antennas. DNNs have shown promise in maximizing the sum rate, but despite their enhanced performance, DNNs come with high computational complexity and substantial energy requirements, posing challenges for deployment on resource-limited devices like edge platforms.

To tackle the complexities and energy demands associated with DNN-based precoders in mMIMO systems, we focus on developing a method that preserves the high throughput of DNNs while making them suitable for resource-constrained environments. This requires overcoming inefficiencies caused by uniformly applied quantization, which overlooks the varying sensitivity of different network layers to precision reduction.

In this work, we introduce a novel framework that combines mixed-precision quantization-aware training with NAS. Our method is designed to significantly lower the energy consumption of DNNs in precoder design while sustaining high throughput for site-specific BSs. Extensive experiments validate that our approach achieves superior energy efficiency compared to existing DL-based and conventional beamforming techniques.

3.4 Methodology

3.4.1 Model Architecture and NAS

The proposed DNN architecture for digital beamforming, illustrated in Figure 3.1, is specifically designed to address the demands of mMIMO systems. The architecture gets the real and imaginary parts of the channel state information through two separate channels as input and predicts the precoding matrix. The network comprises a single convolutional layer with

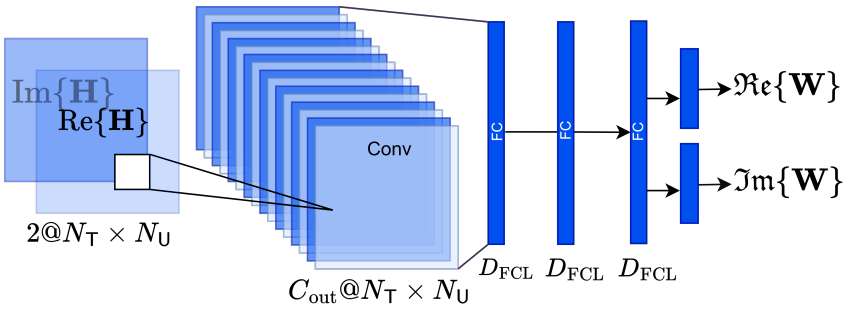


Figure 3.1 DNN architecture template.

the kernel size of 3 for feature extraction, followed by three Fully Connected Layers (FCLs) to map the extracted features into the real and imaginary components of the digital precoder. To enhance model generalization and stability, the architecture incorporates ReLU activation functions, 5% dropout to mitigate overfitting, and batch normalization to accelerate convergence and improve robustness during training.

The design of a CNN-based architecture draws inspiration from [72] and [63], which showcase its effectiveness in hybrid beamforming applications. To balance complexity and performance, preliminary experiments led to a base DNN template featuring one convolutional layer and three FCLs. Opting for a single convolutional layer stems from observations that additional layers offer diminishing improvements in sum rate performance while significantly increasing computational demands. Similarly, three FCLs were selected, as fewer layers compromise performance and additional ones inflate complexity without proportional gains. The size of each layer is configurable via NAS. The energy consumption of our design, assuming 64 output channels for CNNs and 1024 neurons for FCLs, increases by approximately 19% when adding a convolutional layer and by 5% when adding an FCL.

Our approach optimizes quantization by exploring a search space of bit-width configurations for the weights across the layers of the DNNs. We vary the precision levels for each layer—[CONV, FCL1, FCL2, FCL3]—with multiple bit-width choices, allowing for a large number of possible permutations. Additionally, we consider different model architectures by varying the output channel size C_{out} of the convolutional layer and the size D_{FCL} of FCLs. This results in a comprehensive search space combining quantization and architectural configurations. We use exhaustive search to explore all possible combinations, ensuring that optimal configurations are identified without overlooking any potential solutions. The small model size and fast training times make this exhaustive search feasible. Ultimately, this method helps to balance energy efficiency and sum rate performance, which is essential for energy-constrained mMIMO systems where optimizing both performance and resource usage

is critical.

3.4.2 Quantization Methods

This study examines several quantization techniques to enhance energy efficiency and maintain performance in DNNs for mMIMO systems. The main objective is to minimize energy consumption and memory demands while preserving an acceptable sum rate. We explore QAT using the LSQ method [44] and MPQ through an exhaustive search. These quantization methods are then combined with NAS to find an optimal balance between energy efficiency and performance.

PTQ

In preliminary experiments, we also considered PTQ as a simple way to reduce model complexity. This method applies fixed-point quantization to a pre-trained floating-point (FP32) model without requiring retraining. However, in our PTQ experiments, sum rate performance was highly degraded at low resolutions (4 bits or less), and therefore we focus in this paper on the QAT approach to investigate the best possible model compression.

QAT

To address the limitations of PTQ, QAT fine-tunes a pre-trained model within quantization constraints, enabling the network to adapt to lower precision with minimal performance loss. We use LSQ [44], which dynamically adjusts quantization step sizes during training. Unlike static step quantization, LSQ optimizes these step sizes to mitigate quantization error, especially in low-bit scenarios. This is achieved through the Straight Through Estimator [73], which supports gradient-based optimization for non-differentiable quantization functions. The adaptive nature of LSQ provides significant gains in both sum rate and energy efficiency under low-bit quantization constraints.

MPQ with NAS

MPQ assigns different bit-widths to each network layer, offering a flexible trade-off between energy efficiency and accuracy. We employ exhaustive search to evaluate quantization combinations [$2, 4, 8,$ and 16 bits] across four layers, calculated as $4^4 = 256$. and various model architectures by varying $C_{\text{out}} \in \{8,\ 16,\ 32,\ 64\}$ and $D_{\text{FCL}} \in \{512, 1024\}$ across the CNN and FCLs (see Fig. 3.1) to minimize energy consumption while preserving sum rate performance [74].

3.4.3 Training Method

All models are trained using a self-supervised approach [63], with sum-rate maximization as the objective function. This method ensures that the model learns to predict effective precoding vectors directly, without the need for labeled data. The loss function is defined as

$$\mathcal{L} = -R(\mathbf{W}) \,, \tag{3.6}$$

and models are trained with a fixed learning rate of 10^{-3} and a batch size of 1000. Additionally, we average the results from four training runs with distinct initialization seeds to control for variations in the training process.

3.4.4 Energy Model for Quantized Neural Networks

In order to evaluate and compare the energy consumption of a DNN solution, we consider a simple but realistic model of the energy consumed to compute the DNN output. We base our model on the one proposed in [75], which considers the energy required for memory accesses and computations, while taking into account the impact of the bit width of model weights and activations.

The energy consumption of the DNN is thus decomposed into three components: computation energy (E_C), weight transfer energy (E_W), and activation transfer energy (E_A). The total energy consumption is then given by

$$E_{\text{DNN}} = E_C + E_W + E_A \,. \tag{3.7}$$

The computation energy is based on counting the number of MAC operations required for the linear portion of each layer and the number of biasing, batch normalization, and activation function computations. To simplify the model, all these operations are attributed an energy E_{MAC}. The computation energy E_C is thus given by

$$E_C = E_{\text{MAC}} \cdot (N_c + 3 \cdot N_a) \,, \tag{3.8}$$

where N_c is the total number of MAC operations required by the model, N_a is the total number of activations, and the factor 3 arises since one biasing, one normalization, and one activation must be computed for each activation output.

For consistency with the baseline energy model that will be presented next, we model E_{MAC}

in terms of the bit width Q as

$$E_{\text{MAC}} = \alpha \cdot \left(\frac{Q}{16}\right)^{\beta}, \tag{3.9}$$

while choosing $\alpha = 0.86$ and $\beta = 1.9$ to fit the energy measurements reported in [1].

The energy associated with transferring weights, E_W, is expressed as

$$E_W = E_M \cdot N_w + E_L \cdot \frac{N_c}{\sqrt{p}}, \tag{3.10}$$

where N_w is the total number of weights, E_M and E_L represent the energy costs of accessing the main on-chip memory and local buffers, respectively, and p is the number of parallel execution units. We use $E_M = 2E_L = 2E_{\text{MAC}}$ and $p = 64\left(\frac{Q}{16}\right)$. It is assumed that the entire DNN model fits on-chip and no accesses to external memory are needed.

Finally, the energy for transferring activations, E_A, is given by

$$E_A = 2 \cdot E_M \cdot N_a + E_L \cdot \frac{N_c}{\sqrt{p}}\,. \tag{3.11}$$

3.4.5 Energy Model for Baseline Methods

We compare the energy consumption of the proposed quantized DNN models to conventional algorithms, specifically the WMMSE and ZF methods. The WMMSE algorithm, known for its high computational complexity due to iterative processing and matrix inversions, consumes significantly more energy than the ZF approach. Since these energy models act as a baseline, we adopt a conservative (lower bound) approach by accounting only for the multiplications when estimating compute energy. For memory energy estimation, we consider only the local buffer accesses for the operands. The total number of real multiplications required for I iterations of WMMSE is given by

$$\begin{aligned} N_c = I\Bigg(&\frac{8}{3}N_{\mathsf{T}}^3 N_{\mathsf{U}} + 4N_{\mathsf{T}}^2 N_{\mathsf{U}} + 4N_{\mathsf{T}}(4N_{\mathsf{U}}^2 + 2N_{\mathsf{U}}) \\ &+ 4N_{\mathsf{U}}^2 + \frac{56}{3}N_{\mathsf{U}}\Bigg), \end{aligned} \tag{3.12}$$

while the required number of real multiplications for ZF is

$$N_c = 8N_{\mathsf{U}}^2 N_{\mathsf{T}} + \frac{8}{3}N_{\mathsf{U}}^3\,. \tag{3.13}$$

The energy model for WMMSE and ZF can then be expressed as

$$E_{\mathrm{B}} = E_{MAC} \cdot N_c + E_L \cdot \frac{N_c}{\sqrt{p}}\,. \tag{3.14}$$

3.5 Numerical Results

3.5.1 Dataset Definition

A custom dataset was generated to accurately reflect the channel characteristics pertinent to beamforming in mMIMO systems. MATLAB's Ray-Tracing toolbox simulated both Line-Of-Sight (LOS) and Non-Line-Of-Sight (NLOS) conditions. The simulations positioned a BS within the Montreal region, utilizing environmental data from OpenStreetMap [76] for realism. The base station employed a uniform planar array antenna with 8x8 elements, spaced at half-wavelength, operating at a frequency of 2 GHz. The transmitter was set at a height of 20 m and powered at 20 W, assuming a system loss of 10 dB. Users were placed in circular patterns around the base station at distances ranging from 50 to 350 m and at 10-degree intervals. This configuration captured a wide variety of deployment scenarios and channel conditions. The ray-tracing simulations considered up to 10 reflections but excluded diffraction effects, focusing on signal reflections from buildings and terrain to emulate multipath propagation in urban environments accurately.

3.5.2 Energy Consumption Examples

We first present some examples of the energy consumed by the different approaches, as per the model presented in Sections 3.4.4 and 3.4.5. We consider the "UdeM-NLOS" scenario. In Table 3.1, we report the energy for the "default" variants of the method, that is, for WMMSE, we set the stopping criterion to 10^{-5} to have near-optimal performance, and for the DNN, we use $C_{\mathrm{out}} = 64$ and $D_{\mathrm{FCL}} = 1024$ with the maximum weight resolution of 16 bits. We see that the DNN consumes significantly less than WMMSE at the cost of a slight degradation in sum rate (on this scenario). ZF, on the other hand, is much less complex, but does not provide competitive performance in NLOS conditions, or at low Signal-To-Noise Ratio (SNR). As a result, ZF is unlikely to be a favored solution in practice, since it results in significant under-utilization of the BS resources.

Next, to illustrate the impact of quantization, Table 3.2 lists the energy consumption of the DNN for various uniform bit-width configurations, for the same C_{out} and D_{FCL} as in Table 3.1. We see that lowering the weight resolution leads to substantial energy savings but at the cost of a moderate decrease in performance.

Table 3.1 Energy Comparison of the Default Variants on "UdeM-NLOS" ($N_{\mathsf{T}} = 64$, $N_{\mathsf{U}} = 4$, SNR = 15 dB)

Method	**Energy (μJ)**	**Sum Rate (bit/s/Hz)**
WMMSE ($I = 92.8$)	296	19.2
Default DNN	**54.5**	**18.9 ± 0.007**
Zero-Forcing (ZF)	0.008	15.3

Table 3.2 DNN Energy Consumption with Uniform Quantization for $C_{\text{out}} = 64$, $D_{\text{FCL}} = 1024$ on "UdeM-NLOS" ($N_{\mathsf{T}} = 64$, $N_{\mathsf{U}} = 4$, SNR = 15 dB)

Quantization Configs	**Energy (μJ)**	**Sum Rate (bit/s/Hz)**
[16, 16, 16, 16]	54.5	18.9 ± 0.007
[8, 8, 8, 8]	17.5	18.6 ± 0.012
[4, 4, 4, 4]	7.4	17.8 ± 0.014
[2, 2, 2, 2]	4.6	17.1 ± 0.032

3.5.3 NAS Results

Figure 3.2 highlights the trade-offs between computational energy efficiency (bits/s/Hz/μJ) and sum rate (bits/s/Hz) across diverse DNN configurations generated as described in Section 3.4, on the "UdeM-LOS" scenario. Each curve shows the Pareto front corresponding to a particular architecture configuration, while each point in this curve uses a different quantization configuration.

A few trends can be mentioned among the Pareto-optimal results for each architecture. Firstly, the first layer, CONV, is often kept at high precision, particularly in smaller models, to maintain performance while reducing energy consumption. Interestingly, no Pareto-optimal model uses uniform quantization across all layers. Moreover, even models that achieve the highest sum rates do not employ more than two layers at the highest precision. These results emphasize the importance of efficiently distributing bit precision across layers to optimize energy consumption.

The configuration yielding the highest energy efficiency is ($C_{out} = 8$, $D_{FCL} = 512$) with quantization $[2, 8, 8, 8]$, achieving 26.2 bits/s/Hz/μJ at a moderate sum rate of 28.5 bits/s/Hz. On the other hand, the configuration achieving the highest sum rate is ($C_{out} = 64$, $D_{FCL} = 1024$) with quantization $[16, 16, 4, 8]$, which reaches 31.3 bits/s/Hz but does not use full precision, making it more interesting from an energy efficiency perspective. An alternative worth mentioning is the model with the second-best sum rate of 31.1 bits/s/Hz, achieved with an archi-

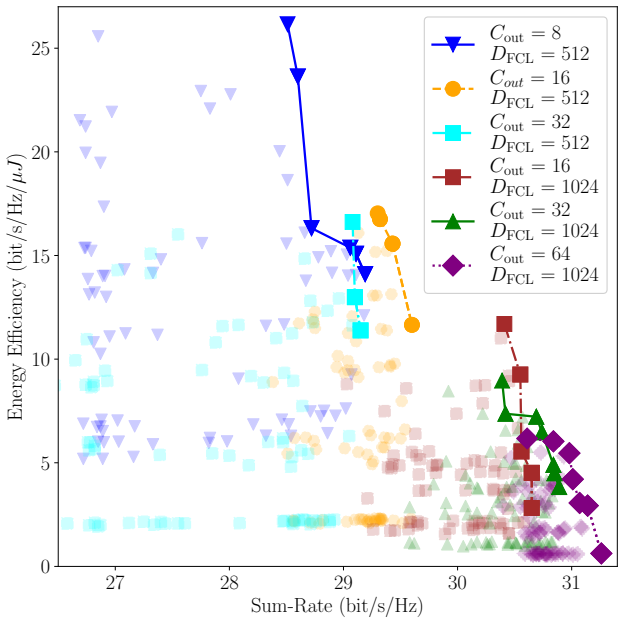


Figure 3.2 Trade-off between energy efficiency and sum rate for CNNs with varying C_{out}, D_{FCL}, and MPQ bit widths, on the "UdeM-LOS" scenario ($N_{\mathsf{T}} = 64$, $N_{\mathsf{U}} = 4$, average SNR = 29 dB). All the model configurations that were evaluated are shown, while the curves provide the Pareto front associated with each architecture configuration.

tecture of ($C_{out} = 64$, $D_{FCL} = 1024$) and quantization $[16, 2, 2, 2]$, which uses nearly 5× less energy than the highest sum-rate model. We do observe compute energy efficiency decreasing rapidly near the highest sum-rate, but of course this could simply mean that switching to a larger and/or different DNN architecture would be preferable at that point.

To further illustrate the importance of NAS and model compression in finding efficient DNN precoders, Figure 3.2 includes horizontal and vertical arrows that quantify the impact on performance of the design choices. The horizontal arrow measures the difference in sum rate between the worst and best configurations at equal energy efficiency, revealing a 20% improvement through optimal model selection. Similarly, the vertical arrow shows the difference in energy efficiency at an equal sum rate, demonstrating a 14.5× gain.

3.5.4 Impact of Deployment Environment and Energy Efficiency Comparison

Figure 3.3 compares energy efficiency (bits/s/Hz/μJ) and sum-rate (bits/s/Hz) across two contrasting deployment scenarios: UdeM-NLOS, characterized by challenging multipath conditions, and Okapark-LOS, offering clear LOS signal propagation. These two scenarios highlight the adaptability and effectiveness of the proposed quantized models. For the DNN precoders, each curve shows the Pareto-optimal configurations across the entire search space,

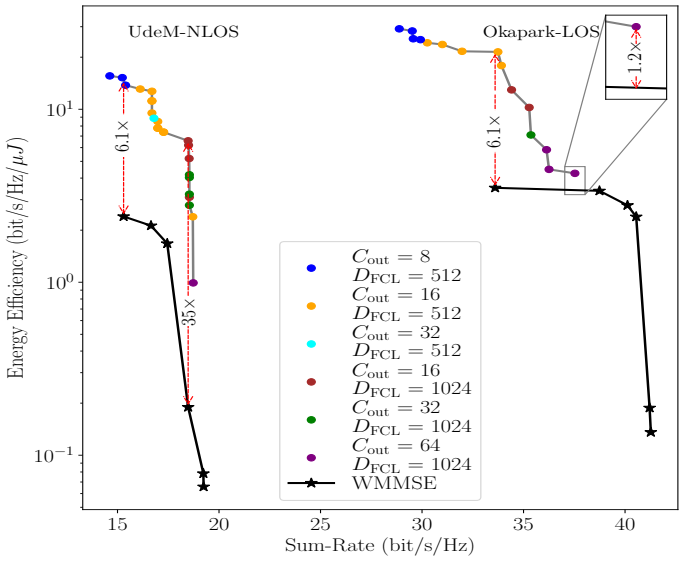


Figure 3.3 Comparison of energy efficiency (bits/s/Hz/μJ) and sum rate (bits/s/Hz) across two environments: UdeM-NLOS (average SNR = 15 dB) and Okapark-LOS (average SNR = 28 dB). The proposed method achieves a superior balance of energy efficiency and sum rate performance compared to WMMSE. Results are derived for models with varying (C_{out}) and (D_{FCL}).

whereas for WMMSE, the trade-off between sum-rate and energy efficiency is varied by adjusting the stopping criterion.

In the UdeM-NLOS environment, sum rates are constrained between 15 and 20 bits/s/Hz due to severe signal attenuation and multipath effects. Despite these limitations, the quantized models achieve significant energy savings, with improvements of up to 35× in energy efficiency compared to the WMMSE baseline, all while maintaining competitive sum rates.

In contrast, the Okapark-LOS environment, which benefits from clear signal paths, supports higher sum rates ranging from about 30 to 40 bits/s/Hz. Depending on the desired sum-rate, the DNN precoders can provide improvements in energy efficiency ranging from 6.1× to 1.2×. However, with the DNN architecture template and training method considered in this paper, the DNN precoder is unable to achieve the highest sum-rate that can be provided by WMMSE.

These results emphasize the adaptability of the quantized models across diverse deployment environments, and their ability to achieve the same performance with less compute energy. Interestingly, the energy gains provided by DNNs appear to be larger in more difficult (low SNR, non line-of-sight) environments.

3.6 Conclusion

We proposed a novel framework for finding efficient compressed DNN models for mMIMO precoding. By using quantization-aware training with LSQ and MPQ, while also searching through different DNN architecture sizes, we find DNN models with significantly lower energy consumption at equal performance. A variety of Pareto-optimal models were generated, showing the ability of DNN solutions to provide different tradeoffs between energy consumption and performance. Compared to WMMSE, the obtained DNN models achieve superior energy efficiency across diverse deployment scenarios. This work demonstrates the promise of DNN solutions to obtain high-performance mMIMO precoders with much improved energy efficiency.

CHAPTER 4 BITADAPT: AN ENERGY-AWARE PRECISION LEARNING FRAMEWORK

4.1 Background & Motivation

DNNs applied to wireless communications tasks such as mMIMO precoding must meet stringent requirements on both prediction accuracy and implementation efficiency. Our prior work in Chapter 3 demonstrated that MPQ combined with NAS can yield compact neural precoders that preserve sum-rate performance while reducing computational cost. However, such search-based methods suffer from several critical drawbacks.

First, the search complexity grows combinatorially: exploring B bit choices across L layers requires $O(B^L)$ evaluations, making it infeasible for deeper networks. For example, a Transformer model with 4 layers and multiple linear projections can exhibit on the order of 10^8 candidate bit configurations. Second, the resulting precision allocation is rigid: any change in wireless environment, target sum-rate, or hardware constraints would demand a new discrete search. Third, such methods offer no gradient signal to guide the learning process, as bit-widths are treated as discrete variables. This prevents leveraging modern optimization techniques such as backpropagation, leading to longer search times and less efficient solutions. Finally, exhaustive methods lack interpretability and adaptability. They do not reveal which layers are most sensitive to quantization, nor do they permit real-time adjustment in response to changing hardware or channel conditions. These limitations strongly motivate the need for a more flexible and scalable alternative. BitAdapt addresses this challenge by introducing an end-to-end differentiable framework that jointly learns network weights, quantization parameters, and layerwise bit-widths. This approach enables fine-grained control over the energy–performance trade-off, and supports generalization across deployment domains.

In this chapter, we introduce *BitAdapt*, a gradient-based, deployment-agnostic precision-learning framework that we apply per site to obtain models adapted to each environment that retain the core insight of BitPruning [12], namely that energy can be reduced by pruning *bits* instead of weights, so the network's topology and information flow remain intact. BitAdapt departs from the original method in two ways. First, it embeds the LSQ method [44], allowing each layer to learn its quantizer step size Δ^l concurrently with the weights, which removes the need for post-training calibration. Second, it augments the training loss with a differentiable, hardware-calibrated energy term that continually nudges bitwidths toward the accuracy–energy Pareto frontier. These modifications let BitAdapt co-optimize weights, step sizes, and bitwidths within a single training loop, eliminating the separate discrete search

phase required by the original BitPruning workflow.

By integrating QAT with LSQ, bit selection, and an energy-aware regularization term, our approach enables a network to co-optimize accuracy and EE in a single training pass. Crucially, BitAdapt does not rely on manually tuning each layer's precision or on exhaustive discrete searches; instead, the network learns to allocate bits where they most benefit performance. As before, we adopt a site-specific design strategy: each model is trained on channel data from a specific deployment site, so that the learned precision configuration is tailored to that environment. While BitAdapt removes the need for a brute-force bitwidth search, it can also be combined with NAS for architectural optimization. For example, one could use NAS to select a network architecture per site (as in Chapter 3) and then apply BitAdapt to fine-tune the layer precisions within that architecture.

Our method builds on LSQ for differentiable quantization. LSQ introduces a trainable quantizer step size Δ per layer, allowing the network to adapt the quantization resolution to the weights' distribution. We further employ the Gumbel-Softmax trick to relax discrete bit choices into a continuous optimization problem: each layer's bitwidth is drawn from a categorical distribution over $\{1, 2, \ldots, 16\}$ bits, parameterized by learnable logits.

During training, the network samples a bitwidth for each layer, quantizes weights with that precision, and backpropagates gradients to update both the weights, the step sizes, and the bitwidth logits. As training progresses, the Gumbel temperature is annealed so that each layer's bitwidth distribution converges to a one-hot choice. In effect, the network learns both how many bits and what quantization step size each layer requires to meet the accuracy target with minimal energy cost.

4.2 Differentiable Bitwidth Optimization

BitAdapt jointly optimizes (i) the network weights W, (ii) the LSQ $\{\Delta^l\}$, and (iii) the integer bit-widths $\{b^l\}$ of every layer in a single end-to-end training loop. We begin with the core quantization mechanism.

4.2.1 Learnable Quantizer via LSQ

For each layer l, let W^l denote its real-valued weight tensor. LSQ introduces a trainable step size Δ^l so that the quantized weights $\hat{W}^l$ are

$$\hat{W}^l = \Big[\mathrm{clip}\Big(\mathrm{round}(W^l/\Delta^l),\, q_l^-,\, q_l^+\Big)\Big]\,\Delta^l, \qquad q_l^- = -2^{\,b^l-1}, \qquad q_l^+ = 2^{\,b^l-1}-1. \tag{4.1}$$

The function $\mathrm{round}(\cdot)$ maps to the nearest integer, while $\mathrm{clip}(x,a,b) = \min\big(\max(x,a),\, b\big)$ ensures the integer code falls inside $[q_l^-, q_l^+]$. Hence, with b^l signed bits the quantizer represents the symmetric range $\big[-2^{\,b^l-1},\, 2^{\,b^l-1}-1\big]$ with level spacing Δ^l.

During each forward pass, we sample an *integer* bit-width $b^l \in \{2,\ldots,8\}$ from a Gumbel–Softmax distribution, compute q_l^-, q_l^+, and quantize W^l via (4.1). Step sizes Δ^l and the full-precision weights remain in floating-point memory and are updated by back-propagation.

Because round and clip are non-differentiable, we adopt the STE, propagating gradients as if both operations were the identity. LSQ further rescales the gradient of Δ^l by multiplying it with g_l for numerical stability. This scaling factor is defined as

$$g_l \;=\; \frac{1}{\sqrt{N_l\, q_l^+}}\,,$$

where $N_l = |W^l|$ is the number of parameters in layer l. This scaling follows Esser *et al.* [44] and has proved crucial for convergence when learning both step size and bit-width simultaneously.

4.3 Energy-Aware Regularization

The LSQ mechanism alone discovers low precision solutions, but it lacks any explicit awareness of hardware cost. To steer the bit allocation towards energy savings, we add a differentiable penalty based on an energy model. For each layer l, the dynamic energy consumed by a given bit-width b^l is modeled as

$$E^l(b^l) \;=\; n^l_{\mathrm{MAC}}\, E_{\mathrm{MAC}}\Big(b^l\Big) \;+\; n^l_{\mathrm{mem}}\, E_{\mathrm{mem}}\Big(b^l\Big), \tag{4.2}$$

where n^l_{MAC} and n^l_{mem} are the MAC and memory-access counts dictated by the layer shape. Following measurements from a 45 nm CMOS technology reported by Horowitz [1], the ele-

mentary energy constants are set to

$$E_{\mathrm{MAC}}(b) \;\approx\; 0.86\Big(\tfrac{b}{16}\Big)^{1.9}\ \mathrm{pJ}, \qquad E_{\mathrm{mem}}(b) \;\approx\; 0.43\,b\ \mathrm{pJ}\,.$$

We note that $E^l(b^l)$ is approximately quadratic in b^l for compute and linear in b^l for memory, which agrees with standard architectures for these circuits.

We embed (4.2) into the training objective:

$$\mathcal{L}_{\mathrm{total}} \;=\; \mathcal{L}_{\mathrm{task}} \;+\; \lambda \sum_{l} E^l\Big(b^l\Big), \tag{4.3}$$

where $\mathcal{L}_{\mathrm{task}}$ is the negative sum-rate achieved by the mMIMO precoder and λ balances accuracy against energy. During gradient descent, each layer reduces its bit-width whenever a drop in $E^l(b^l)$ outweighs the rise in $\mathcal{L}_{\mathrm{task}}$. Conversely, precision is preserved where it is critical for performance. Sweeping λ therefore traces an accuracy–energy Pareto frontier without manual bit tuning.

The procedure is architecture-agnostic: in both our CNN and Transformer model precoders, BitAdapt learns site-specific mixed-precision maps that match local channel statistics, yet requires no human intervention. Separate models are trained for every measurement site (as in Chapter 3), ensuring each BitAdapt solution is fully specialized to its deployment environment.

4.4 Training Protocol

The goal of training is to co-optimize three distinct yet tightly interacting sets of variables: network weights W, LSQ step sizes $\{\Delta^l\}$ and per-layer bit-widths $\{b^l\}$ under a single objective that trades SE against hardware energy. To clarify how these elements evolve in concert, this section proceeds chronologically from data ingestion to the final export step, pausing to explain the design choices that proved essential for stable optimization.

4.4.1 Stage 1: Data Preparation and Normalization

Each experiment is strictly site-specific: a model trained for one deployment cell is never reused in another location. The raw dataset, described in Section 3.5.1, comprises complex-valued channel matrices $\mathbf{H} \in \mathbb{C}^{N_{\mathrm{U}} \times N_{\mathrm{T}}}$, generated using MATLAB's Ray-Tracing toolbox under diverse LOS and NLOS conditions. Each matrix is split into its real and imaginary components and concatenated along the antenna dimension to form real-valued inputs of size

$2N_{\mathsf{T}}$ per user.

To reduce the dynamic range of input magnitudes caused by path loss and fading, we apply ℓ_2 normalization across the dataset, ensuring that all channel vectors have comparable scale. While the energy model does not depend on activation values, this normalization step stabilizes the training process by preventing the LSQ quantizer from collapsing to ineffective step sizes early in training, a failure mode observed when operating on unnormalized inputs.

After normalization, the dataset is shuffled and partitioned into training, validation, and test splits with an 80/10/10 ratio. These partitions remain fixed across all experiments to enable fair comparisons. A mini-batch size of 1000 is used consistently, providing stable batch normalization statistics and accurate per-batch energy estimates.

4.4.2 Stage 2: Deterministic Bit-width Selection

Unlike earlier work that samples bit-widths from a categorical distribution, BitAdapt assigns each layer a single real-valued precision parameter $\tilde{b}^l \in \mathbb{R}$, initialized to 8. At run time, we convert it to an integer bit-width via a saturating round-and-clip operation

$$b^l \;=\; \mathrm{clip}\Big(\mathrm{round}(\tilde{b}^l),\, 1,\, 16\Big), \qquad \text{where } \mathrm{clip}(x,1,16) \;=\; \min\{\max\{x,1\},16\}.$$

The clipping step guarantees $b^l \in \{1,\ldots,16\}$ even if $\tilde{b}^l$ drifts outside that range during optimization. Using b^l, we set $q_l^{\pm} = \pm(2^{\,b^l-1}-1)$ and quantize the weight tensor W^l with (4.1). Both the rounding and clipping are handled with the STE, so gradients pass unimpeded to $\tilde{b}^l$. In practice this deterministic, bounded parameterization improves training stability, eliminates the need for categorical sampling, and lets BitAdapt learn precisions on a *continuous* scale while enforcing valid bit-widths at inference time. In practice, this choice delivers three advantages:

1. Stable energy signals. Each forward pass executes with a well-defined precision map, which in turn produces a deterministic analytic energy number. The optimizer therefore sees smooth rather than noisy gradients of the energy term.

2. Fast convergence. In pilot studies, the deterministic scheme achieved the same final rate energy trade-off 20–25 % faster than its stochastic counterpart because the optimizer never "wasted" iterations exploring sub-optimal bit configurations.

4.4.3 Stage 3: Loss Formulation and Gradient Flow

The composite loss used for training corresponds exactly to the formulation in Equation (4.3), where $\mathcal{L}_{\text{task}}$ is instantiated as the negative sum-rate achieved by the precoder output, and $E^l(b^l)$ denotes the calibrated energy surrogate introduced in Section 4.3. The coefficient λ is fixed to 10^{-2} for all deployment sites and both neural architectures. extensive sweeps showed that this single setting captures the knee of the Pareto curve, where large energy savings can be achieved for only a marginal rate loss. Gradients are taken concerning weights, step sizes, and continuous bit-width variables. The round and clip operations adopt the STE, while LSQ rescales $\partial\mathcal{L}/\partial\Delta^l$ by $g_l = 1/\sqrt{N_l q_l^+}$ as recommended in [44]. We clip the global gradient norm to $\|g\|_2 \leq 1.0$ to avoid occasional precision "collapses" in early epochs that would otherwise thrust several layers directly from 8-bit to 1-bit in a single update step.

4.4.4 Stage 4: Optimizer Configuration and Epoch Schedule

The shallow convolutional precoder is optimized with Adam (base learning-rate 1×10^{-3}, weight decay 1×10^{-5}), whereas the deeper Transformer model uses AdamW (base learning-rate 1×10^{-4}, weight decay 1×10^{-3}). Both architectures share an auxiliary parameter group comprising all $\tilde{b}^l$ values; that group uses Adam/AdamW but with a larger *dedicated* Learning Rate (LR) of 5×10^{-4} so that precision parameters adapt slightly faster than weights or step sizes. A constant rate over **200 epochs** consistently delivered the best SR-EE trade-off and the most predictable training curve.

4.4.5 Stage 5: Validation and Checkpointing

At the end of every epoch, we evaluate the network on the fixed 10 % validation split, record the sum-rate R_{val}, the analytic network energy E_{val}, and their ratio $\text{EE}_{\text{val}} = R_{\text{val}}/E_{\text{val}}$. Because energy is computed deterministically, EE_{val} traces a smooth, monotone curve and typically plateaus long before the final epoch. Nevertheless, the optimizer always completes the full **200 epochs**. When training ends, we scan the saved logs, identify the epoch whose checkpoint maximises EE_{val}, and export only that model for downstream analysis. The run length is fixed in advance and identical across all experiments.

4.4.6 Stage 6: Deployment and Hardware Export

After training, each continuous precision is rounded once more:

$$b^l_{\text{deploy}} \;=\; \text{round}\Big(\tilde{b}^{\,l}\Big),$$

and stored as an unsigned integer in the model header. The corresponding LSQ step size Δ^l is fixed to its final floating-point value and subsequently quantized to a 16-bit fixed-point representation that matches the target accelerator's format. All normalization layers, together with the final softmax normalization in the transformer, remain in FP32 because their energy footprint is three orders of magnitude smaller than that of convolutions, attention projections, or fully connected layers. The export script therefore produces a deterministic, hardware-ready MPQ network whose bit-width map is (i) energy-aware, (ii) fully differentiable during training, and (iii) tailored to the local channel statistics of a single physical deployment site.

4.5 Architecture–Specific Details

BitAdapt treats "number of bits" as just another learnable parameter, so in principle it works for any quantizable layer. In our evaluation, we apply it to two precoders chosen to span the design space one shallow, convolutional network aimed at low-footprint edge devices, and one deeper Transformer that pushes sum-rate performance by modelling long-range channel structure. In both cases, every weight tensor owns its precision variable; the associated activations are quantized with the same bit-width, so the data-path width remains uniform in hardware.

4.5.1 CNN Precoder

The convolutional model starts with a single 3×3 layer (C_{in}=2, C_{out}=64) followed by three fully connected stages of width $\{1024, 512, N_{\mathsf{T}}N_{\mathsf{U}}\}$. Four real-valued precision scalars $\tilde{b}^l$, one per layer, are learned jointly with the weights and LSQ step sizes. During optimization, the energy term is evaluated with layer-wise MAC and memory counts (as detailed in Chapter 3) but no extra hyper-parameters are introduced. BitAdapt consistently converges to a pattern in which the first convolution and the largest hidden layer keep 6–8 bits, while the penultimate and output layers drop to 3–4 bits. This allocation minimizes energy at an identical sum-rate because errors are injected early in the pipeline propagation, whereas late-stage errors are largely absorbed by the final power-normalization step.

4.5.2 Transformer Precoder

Our second architecture is based on a foundation Transformer model introduced in [77]. At a high level, the model processes the $N_{\mathsf{U}} \times N_{\mathsf{T}}$ channel matrix $\mathbf{H}$ by dividing it into N_{U} input tokens, where each token corresponds to the CSI vector of a single user. Additionally, a dedicated context token is included to encode the users' rate targets $\{R_u^\star\}$. This complete

sequence of tokens is then passed through a Transformer encoder.

The encoder consists of a multi-head self-attention (MHSA) module with a model dimension of $d = 128$ and $h = 4$ attention heads. This is followed by a two-layer feed-forward network based on the SwiGLU activation function, featuring a hidden width of $d_{\text{hid}} = 2048$. Each Transformer block therefore contains six trainable weight matrices: W_Q, W_K, W_V, and W_O for the attention mechanism, along with W_{up} and W_{down} for the feed-forward layers. A full stack of four Transformer blocks results in a total of 24 trainable tensors.

In line with the method proposed in [77], we branch the shared feature extractor into two lightweight output heads. The first head predicts the complex-valued precoding matrix $\widehat{\mathbf{W}}$, while the second estimates an energy-control tuple $(\widehat{\mathbf{\Omega}}, \widehat{\gamma})$, which determines the set of active antennas and the overall scaling of the transmit power.

Integration with BitAdapt Each weight matrix in the Transformer architecture is assigned its own precision variable b^l and LSQ step size Δ^l, following the same procedure used in the CNN-based design. During training, BitAdapt is exposed to the full analytical MAC count of the attention–feed-forward stack, encompassing the query, key, value, and output projections, as well as both passes through the feed-forward network. Importantly, no Transformer-specific heuristics are enforced, allowing the optimizer to autonomously determine the most effective bit allocation across the architecture.

A consistent pattern emerges across different deployment sites. The projections for queries, keys, and values in the first Transformer block tend to stabilize around 7 to 8 bits, preserving the dynamic range necessary for accurate softmax computation. In contrast, attention layers deeper in the stack typically drop to lower precisions around 3 to 5 bits without incurring noticeable degradation in rate performance. The large matrices within the feed-forward networks generally settle between 4 and 6 bits, reflecting a balance between their substantial contribution to MAC energy and their influence on SE.

4.6 Numerical Results

This section provides an overview of the experimental framework used throughout this numerical study. We consider two wireless propagation scenarios, deliberately selected to represent distinct propagation conditions:

- **Ericsson Site:** This scenario models a typical urban macro-cell environment characterized by rich scattering and mixed LOS/NLOS propagation, representative of densely built-up metropolitan areas.

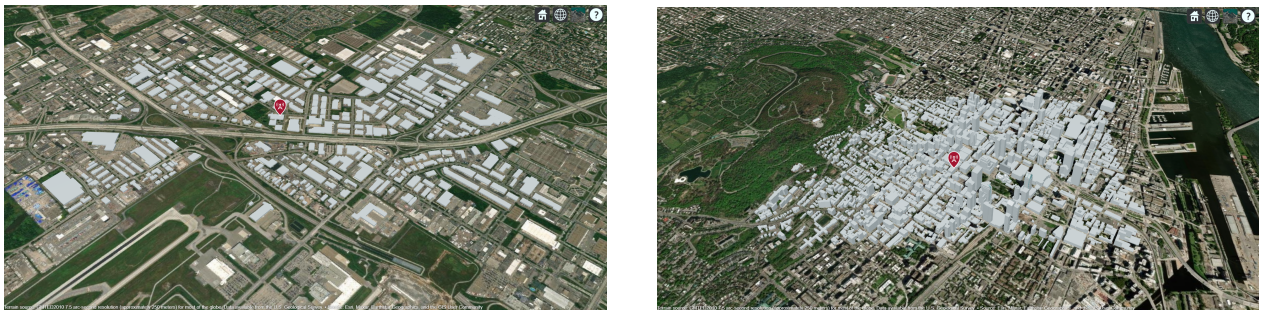
- **Sainte-Catherine Site:** This scenario corresponds to an urban micro-cell with shorter link distances and stronger blockage variability, exhibiting mixed LOS/NLOS propagation with more limited scattering.

For each of these scenarios, we train Transformer model-based DL models across six distinct architectures (footprints). These architectures systematically span from the smallest configuration (**Tiny**: 2 Transformer blocks, 2 attention heads, and approximately 0.2 M parameters) to the largest tested model (**Extra-Large**: 8 Transformer blocks, 8 attention heads, and roughly 18.6 M parameters). This comprehensive range is intended to rigorously investigate the trade-offs between model complexity, sum-rate, and EE.

All Transformer models are optimized according to a unified energy–accuracy loss function given by eq. (4.3). As a classical baseline, we employ the WMMSE precoder, computed with a stopping criterion tolerance of 10^{-5}. Performance evaluation across all scenarios and architectures employs three primary metrics: EE, SR, and bit precision.

4.6.1 Influence of EE-SR Trade-off Coefficient (Hyperparameter Exploration)

To analyze the influence of the energy-accuracy trade-off coefficient, γ, we systematically varied its value over four orders of magnitude during BitAdapt training at the Sainte-Catherine site. Figure 4.1 shows 3D renders of the two deployment sites used to generate our ray-traced datasets and to evaluate BitAdapt: (a) the Ericsson industrial campus and (b) the Sainte-Catherine downtown area in Montréal. Specifically, we considered γ values ranging from 10^{-3} to 10^{2}, as illustrated in Figure 4.2. Each experiment utilized identical data partitions, optimizer configurations, and initial model parameters, ensuring that observed performance variations were solely attributable to changes in the weighting of the energy component within the loss function.



(a) Ericsson Site (b) Sainte-Catherine Site

Figure 4.1 3D renders of the two deployment sites used for dataset generation and evaluation.

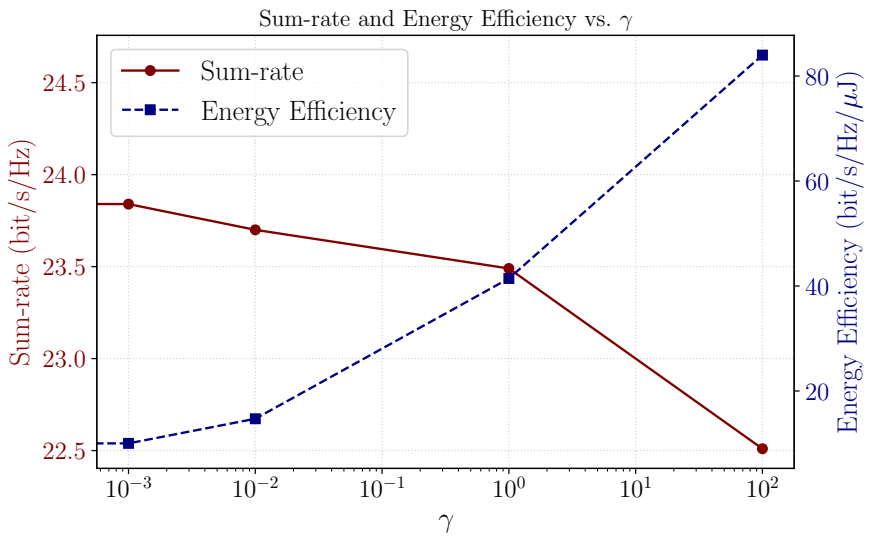


Figure 4.2 Impact of the energy-accuracy trade-off coefficient (γ) on BitAdapt training performance for the Sainte-Catherine site. The maroon curve (left axis) represents the downlink sum-rate (bit/s/Hz), while the blue curve (right axis) indicates EE (bit/s/Hz/µJ), computed via the analytic model detailed in Chapter 4. Each marker denotes the performance obtained from independent training runs corresponding to the indicated value of γ.

Figure 4.2 presents two critical performance metrics as functions of γ: the downlink sum-rate (maroon curve, left y-axis) and the corresponding EE (blue curve, right y-axis). The EE metric (in bits/s/Hz/μJ) was calculated using the analytical energy model detailed in Chapter 3. The horizontal axis employs a logarithmic scale to clearly demonstrate performance trends across different magnitudes of γ.

At smaller values of γ (10^{-3} and 10^{-2}), the loss function predominantly emphasizes the accuracy component (SR). Consequently, BitAdapt maintains relatively high bit-width precision across all layers, resulting in higher overall energy consumption and correspondingly low EE. As γ increases to the intermediate range (10^{-2} to 1), BitAdapt begins reducing precision selectively in layers less sensitive to quantization errors. This precision adjustment markedly decreases energy consumption, significantly improving EE without causing substantial deterioration in sum-rate performance, which remains almost constant throughout this interval.

However, further increasing γ (up to 10^{2}) overly emphasizes energy minimization, forcing most layers toward their lowest permissible precision. While this aggressively quantized configuration maximizes EE, it noticeably reduces the sum-rate performance, making such extreme settings viable only in contexts with stringent power constraints where EE critically outweighs spectral efficiency objectives.

The analysis identifies a practical and optimal **knee region** in the trade-off curve. Moderate γ values, particularly around 10^{-2}, effectively balance significant EE improvements with minimal sum-rate loss. Consequently, for subsequent experiments, we adopt $\gamma = 10^{-2}$, a value representing an optimal trade-off that achieves substantial energy savings without significantly impacting spectral efficiency.

4.6.2 EE-SR Trade-offs NAS

Figure 4.3 illustrates how model capacity, adaptive precision, and the loss trade-off coefficient λ jointly shape the EE–SR landscape. Larger Transformer backbones reach higher downlink sum-rates, as seen in Figure 4.3b, but they also consume markedly more inference energy, 4.3c. Because energy grows faster than the rate benefit, Figure 4.3a shows a consistent drop in energy efficiency (bits/s/Hz/µJ) as model size increases, mirroring the cross-model Pareto trends reported earlier in Figure 3.2.

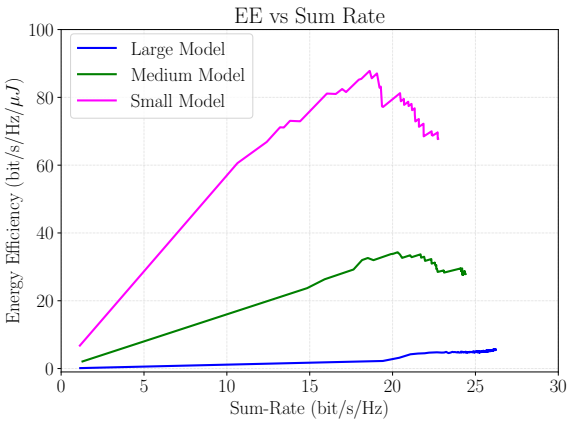
The evolution of the curves also reveals BitAdapt's two-phase quantization strategy. During the first few epochs, the scheduler slashes bit-widths in layers that tolerate coarse precision, producing an abrupt energy drop and a steep vertical jump in EE. Once a low-energy baseline is established, training enters a refinement stage in which precision is selectively restored for performance-critical layers, nudging the trajectories rightwards until further bit-width gains become energetically unjustified.

Adjusting the coefficient λ tilts this balance: higher values stop the trajectories sooner, favoring efficiency over rate, whereas lower values extend them toward higher rates at reduced EE, a pattern that echoes the sweep observed in Figure 4.2. Altogether, the figure underscores that BitAdapt's learned layer-wise precision allocation is pivotal for navigating the EE–SR Pareto frontier; smaller models keep early layers nearly full-precision to guard accuracy, while larger ones aggressively quantize non-critical layers and reserve their bit budget where it yields the greatest sum-rate benefit.

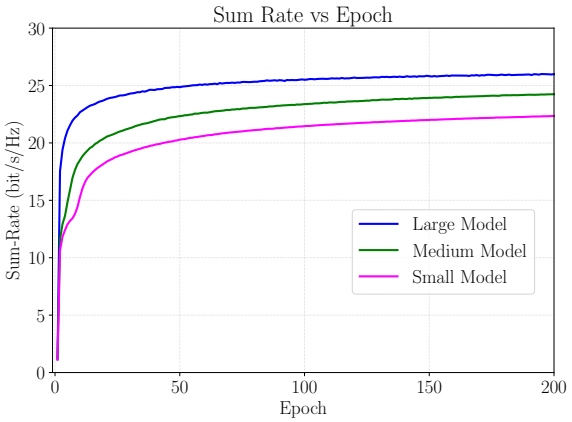
4.6.3 Layer-wise Precision Allocation (Detailed Mechanism Analysis)

Figure 4.4 dissects the weight bit-widths learnt by BitAdapt on the Sainte-Catherine cell under two extreme operating targets: (i) *maximum sum-rate* (violet bars) and (ii) *maximum EE* (green bars). The dashed vertical line at 16 bits marks the full-precision bit-widths baseline.

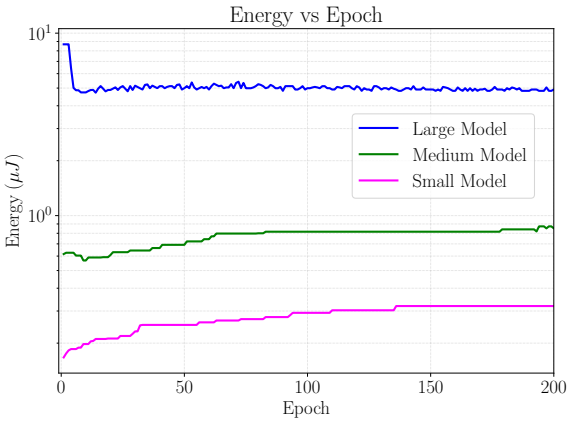
A clear pattern emerges. The *high-SR* model sustains medium precisions ($\sim$9–13 bits) across attention and feed-forward stacks, escalating to $\sim$15 bits in the penultimate block that feeds



(a) EE vs Sum-Rate



(b) Sum-Rate vs Epoch



(c) Energy vs Epoch

Figure 4.3 BitAdapt training on the Sainte-Catherine site for Transformer precoders of three sizes. Panel (a) traces the EE–SR trade-off, (b) shows SR convergence, and (c) tracks inference energy over 200 epochs. All runs share the same loss weight λ; observed gaps arise solely from model size and BitAdapt's adaptive-precision scheduling.

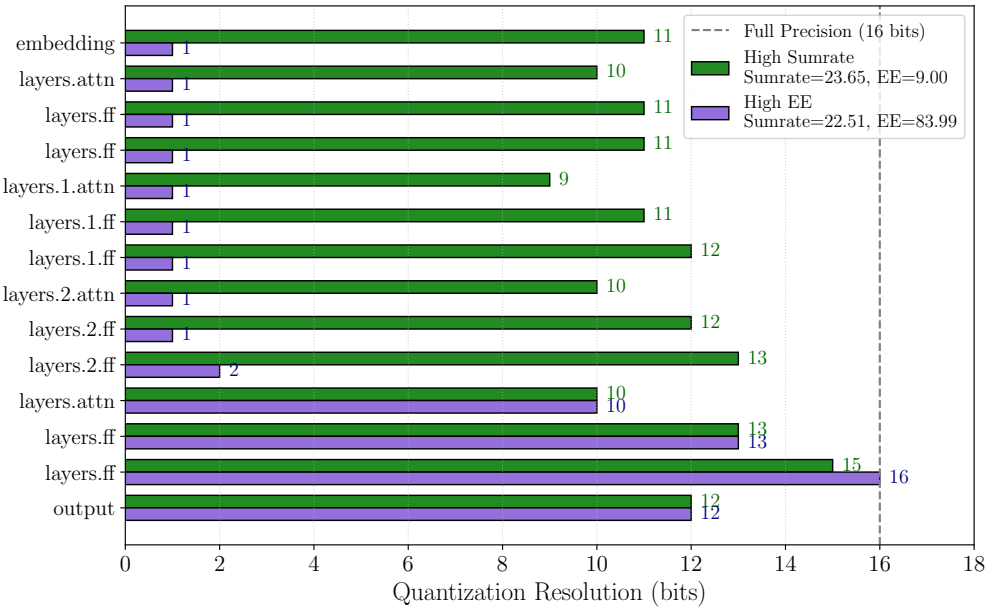


Figure 4.4 Layer-wise bit-widths selected by BitAdapt on Sainte-Catherine for two objectives: **violet** for high sum-rate, and **green** for high EE. The dashed line marks 16-bit precision. BitAdapt aggressively reduces most internal layers to 1–2 bits in the energy-focused setting, while retaining higher precision (9–15 bits) in output layers, yielding a $> 9\times$ efficiency gain at only $\approx 8.7\%$ SR loss.

the output projection. This broad dynamic range underpins the best observed throughput ($R = 24.65\,\text{bit/s/Hz}$) at the expense of low EE ($\text{EE} = 9.0\,\text{bit/s/Hz/µJ}$).

Conversely, the *high-EE* configuration drives almost all early and intermediate layers to the minimum allowed resolution (1–2 bits). Only the final three layers, where quantization noise most directly impairs the radiated waveform, retain elevated precisions of ∼9–15 bits. This "bit funnel" cuts analytic energy dramatically, lifting EE to $83.99\,\text{bit/s/Hz/µJ}$ a $> 9\times$ improvement while sacrificing just 8.7% of the peak sum-rate.

Practical Insights BitAdapt automatically channels precision to layers that most influence the end-to-end objective, obviating manual sensitivity studies. Moreover, substantial energy savings are achievable even in large Transformer precoders provided that adequate precision is reserved for the final projection stages. Notably, these precision maps arise within a single differentiable training loop, without any discrete search over bit patterns, reinforcing BitAdapt's deployment practicality.

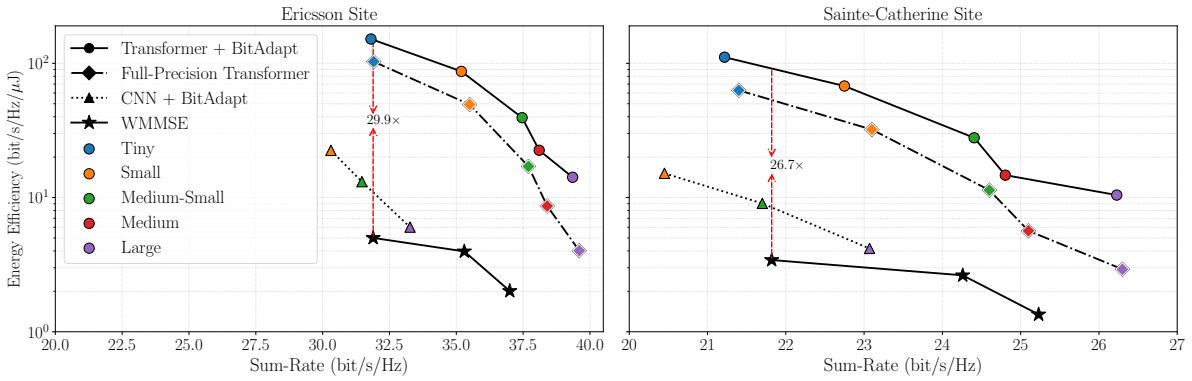


Figure 4.5 EE–SR Pareto fronts at the Ericsson (left) and Sainte-Catherine (right) sites for WMMSE (★), full-precision Transformers (⧫), CNN + BitAdapt (▲), and Transformer + BitAdapt (●). Colored markers denote model sizes from Tiny to Large. Transformer + BitAdapt consistently yields the best trade-off, with up to 29.9× and 26.7× energy efficiency gains over WMMSE.

4.6.4 Site-Specific EE-SR Trade-offs (Comparative Analysis)

Figure 4.5 illustrates the EE–SR Pareto frontiers for four precoding strategies across two deployment sites: Ericsson (left) and Sainte-Catherine (right). The evaluated methods include (★) classical WMMSE, (⧫) full-precision Transformers, (▲) CNN-based BitAdapt, and (●) Transformer-based BitAdapt. Each neural architecture is evaluated under five increasing complexities, ranging from Tiny to Large.

For the CNN architecture, model scaling is governed by the number of output channels $C_{\text{out}} \in \{16, 32, 64\}$ and fully connected layer widths $D_{\text{FCL}} \in \{512, 1024\}$, with the resulting models compressed using BitAdapt to produce the (▲) CNN+BitAdapt Pareto front. The (⧫) Transformer configuration reflects uncompressed full-precision inference, while the (●) Transformer+BitAdapt curve corresponds to NAS-guided, per-layer MPQ models optimized for joint energy and throughput performance.

The WMMSE baseline is computed by systematically varying the convergence tolerance of the iterative solver. Fewer inner iterations reduce computational cost but also degrade sum-rate, thereby sweeping out a continuous EE–SR trade-off. All energy figures follow the calibrated cost model detailed in Chapter 3.4.5. For completeness, we also evaluated the classical ZF precoder. On the Sainte-Catherine site, ZF attains only SR = 13.23 bit/s/Hz with EE = 1653.75 bit/s/Hz/μJ, while at Ericsson it reaches SR = 32.48 bit/s/Hz with EE = 4060.01 bit/s/Hz/μJ. These values confirm the well-known behavior of ZF: it can provide reasonable throughput in favorable line-of-sight conditions but degrades rapidly under practical, interference-limited or correlated channels. The ZF results are not included in

Figure 4.5 in order to zoom-in on the methods achieving high sum rate.

The results reveal several critical insights. First, BitAdapt consistently enables steep energy gains over classical methods: at the Ericsson site, the Tiny Transformer model delivers a 29.9× improvement in energy efficiency compared to WMMSE at similar sum-rate levels, while at Sainte-Catherine, the Tiny CNN variant achieves a 26.7× gain. These margins validate the effectiveness of adaptive precision allocation in reducing energy consumption without degrading throughput.

Second, all Transformer and CNN variants exhibit clean, monotonic Pareto frontiers, reflecting well-structured scaling behavior. As model size increases, both sum-rate and energy consumption rise accordingly. BitAdapt captures this trade-off effectively, allowing models to operate near their energy-optimal point through fine-grained, layer-wise bit-width control.

Third, among all configurations, Transformer+BitAdapt forms the upper bound of performance, consistently dominating other baselines. In several instances, these mixed-precision models even outperform their full-precision counterparts, highlighting the synergistic effect of NAS-guided architecture design and precision-aware quantization.

Finally, the method generalizes robustly across deployment sites. Despite differing propagation conditions, BitAdapt maintains strong performance at both locations without requiring site-specific retraining, underscoring its adaptability and practicality.

In summary, BitAdapt advances the state of the art in energy-efficient precoding by pushing the EE–SR Pareto frontier across architectures and environments. Its combination of adaptive quantization, energy awareness, and architectural flexibility makes it a compelling solution for scalable, real-world deployment in next-generation mMIMO systems.

4.7 Summary of Key Findings

Building on the six Transformer-based architectures evaluated across the Ericsson (urban macro-cell) and Sainte-Catherine (open micro-cell) measurement sites, spanning model footprints from **Tiny** (0.2 M parameters) to **Extra-Large** (18.6 M parameters), this chapter distills the principal insights gained from our EE–SR study of BitAdapt.

BitAdapt achieves **up to 26 ×** higher EE compared to classical WMMSE solutions at comparable sum-rate levels and consistently outperforms fixed-precision baselines. The trade-off between energy and accuracy is tunable via a scalar hyperparameter (γ), with an optimal "knee" region that enables substantial energy savings without degrading throughput. Differentiable bit-width learning allows each Transformer layer to autonomously allocate precision based on sensitivity, resulting in efficient "bit funnel" configurations that prioritize critical

layers. Furthermore, BitAdapt generalizes across deployment sites with differing channel statistics, enabling zero-shot deployment without the need for retraining or architecture redesign.

4.8 Conclusion

Leveraging the empirical evidence detailed in Section 4.6, this chapter has established that BitAdapt transforms Transformer model precoders into truly deployment-ready engines for mMIMO systems. By embedding LSQ-based quantization, continuous bit-width variables and a calibrated energy surrogate directly into the training loss, BitAdapt learns in a single back propagation loop the optimal combination of weights, step sizes, and per-layer precisions. The outcome is decisive: inference energy drops by up to an order of magnitude relative to a uniform 8-bit baseline and by as much as $26.1\times$ when compared with the classical WMMSE precoder, yet the network sustains virtually the same sum-rate. These gains arise from the framework's ability to funnel precision towards the few layers that govern SE, while aggressively quantizing the rest, and to repeat this process afresh for every deployment site so that the final bit map is always matched to local channel statistics and hardware constraints. In short, BitAdapt closes the gap between DL precoding theory and the power budgets of real BS hardware.

CHAPTER 5 CONCLUSION

5.1 Summary of Works

This thesis explored the design and deployment of energy-efficient, site-specific deep neural networks for massive MIMO precoding. The work was structured in two main parts. In the first part, we introduced a mixed-precision quantization framework based on exhaustive search and Neural Architecture Search (NAS), enabling performance-optimized compression tailored to varying channel environments. This approach demonstrated strong energy-performance trade-offs but suffered from scalability limitations and reliance on discrete search mechanisms.

To address these limitations, the second part of the thesis proposed a BitAdapt framework, a differentiable approach for layer-wise precision optimization. By combining Quantization-Aware Training (QAT), Learned Step Size Quantization (LSQ), and an energy-aware regularization term, our method learns bitwidths as part of the training process. The approach was validated across both CNN and Transformer-based architectures, achieving strong Pareto efficiency without the need for costly search procedures. The BitPruning framework successfully eliminated unnecessary bits while preserving full network connectivity, yielding significant improvements in energy efficiency with minimal loss in sum-rate performance.

5.2 Limitations

While BitAdapt demonstrates strong results across two real-world sites, further validation across additional channel conditions and hardware energy models (e.g., FPGA or ASIC-specific estimations) would strengthen its applicability. Additionally, extending BitAdapt to jointly optimize phase resolution in hybrid beamforming or to support dynamic power budgets in real-time scheduling scenarios offers promising directions for future research.

Overall, BitAdapt represents a significant step toward practical, energy-aware, and intelligent precoding in future wireless communication systems.

5.3 Future Research

Future work can proceed along several complementary directions. A priority is hardware in the loop training in which real power monitors replace analytic surrogates, thereby closing the modelling gap for particular FPGA or ASIC fabrics. Another important direction is to assess

robustness under imperfect CSI, where noisy channel estimates or other impairments may degrade performance. BitAdapt could be fine-tuned directly on noisy CSI during training to ensure that compressed models remain reliable under practical conditions. A third avenue is a full network mixed-precision design in which activation tensors are quantized alongside weights, possibly with layer or even channel-wise precision to avoid accuracy loss. Joint analog–digital co-design, where BitAdapt learns both digital bit-widths and quantized RF phase-shifter settings, could further reduce overall power consumption. Online adaptive precision that tracks instantaneous traffic load, channel quality, or battery state, as well as extensions to coordinated multi-point and cell-free scenarios, also offer promising returns. Finally, coupling the optimization with meta-learning may improve robustness when the deployment environment or hardware platform deviates from training conditions.

By addressing these challenges, future research can transform the principles demonstrated in this thesis into deployable solutions for sustainable, high-performance wireless infrastructure.

REFERENCES

[1] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 2014, pp. 10–14.

[2] T. Kebede, Y. Wondie, J. Steinbrunn, H. B. Kassa, and K. T. Kornegay, "Precoding and beamforming techniques in mmwave-massive MIMO: Performance assessment," *IEEE Access*, vol. 10, pp. 16 365–16 387, 2022.

[3] "Mobile industry impact on energy consumption and carbon emissions," GSMA Intelligence, Tech. Rep., 2021, available: https://www.gsma.com/.

[4] M. Usama and M. Erol-Kantarci, "A survey on recent trends and open issues in energy efficiency of 5G," *Sensors*, vol. 19, no. 14, p. 3126, 2019.

[5] T. L. Marzetta, "Noncooperative cellular wireless with unlimited numbers of base station antennas," *IEEE Transactions on Wireless Communications*, vol. 9, no. 11, pp. 3590–3600, Nov. 2010.

[6] E. Björnson, J. Hoydis, and L. Sanguinetti, "Massive MIMO systems with non-ideal hardware: Energy efficiency, estimation, and capacity limits," *IEEE Transactions on Information Theory*, vol. 61, no. 10, pp. 5334–5360, 2015.

[7] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "An iteratively weighted MMSE approach to distributed sum-utility maximization for a MIMO interfering broadcast channel," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4331–4340, 2011.

[8] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management," in *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, 2017, pp. 1–6.

[9] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[10] R. Krishnamoorthi, "Quantizing deep convolutional networks for efficient inference: A whitepaper," *arXiv preprint arXiv:1806.08342*, 2018.

[11] M. Rakka, M. E. Fouda, P. Khargonekar, and F. Kurdahi, "Mixed-precision neural networks: A survey," *arXiv preprint arXiv:2208.06064*, 2022.

[12] M. Nikolić, H. Kwon, K. Choi, T. Hwang, W. Kim, and H.-J. Yoo, "Bitpruning: Learning bitlengths for aggressive and accurate quantization," *arXiv preprint arXiv:2002.03090*, 2020.

[13] A. Alkhateeb, "DeepMIMO: A generic deep learning dataset for millimeter wave and massive MIMO applications," *arXiv preprint arXiv:1902.06435*, 2019.

[14] G. Kasalaee, A. H. Karkan, J.-F. Frigon, and F. Leduc-Primeau, "Compression of site-specific deep neural networks for massive MIMO precoding," *IEEE Int. Conf. on Machine Learning for Communication and Networking (ICMLCN)*, 2025.

[15] T. L. Marzetta, "Massive MIMO: An introduction," *Bell Labs Technical Journal*, vol. 20, pp. 11–12, 2015.

[16] X. Zhao, S. Lu, Q. Shi, and Z.-Q. Luo, "Rethinking wmmse: Can its complexity scale linearly with the number of bs antennas?" *IEEE Transactions on Signal Processing*, vol. 71, p. 433–446, 2023. [Online]. Available: http://dx.doi.org/10.1109/TSP.2023.3244104

[17] B. Liu, H. Liu, and S. Roy, "Structured dirty paper coding with known interference structure at receiver," in *Conference Record of the Thirty-Ninth Asilomar Conference onSignals, Systems and Computers, 2005.*, 2005, pp. 643–647.

[18] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 186–195, 2014.

[19] Y.-T. Chang, Z.-W. Ou, H. Alsuraisry, A. Sayed, and H.-C. Lu, "A 28-GHz Low-Power Vector-Sum Phase Shifter Using Biphase Modulator and Current Reused Technique," *IEEE Microwave and Wireless Components Letters*, vol. 28, no. 11, pp. 1014–1016, 2018.

[20] S. Akhila and Hemavathi, "5G ultra-reliable low-latency communication: Use cases, concepts and challenges," in *2023 10th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2023, pp. 53–58.

[21] A. M. Elbir and A. Papazafeiropoulos, "Hybrid precoding for multi-user millimeter wave massive MIMO systems: A deep learning approach," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 552–563, 2020.

[22] H. Huang, Y. Song, J. Yang, G. Gui, and F. Adachi, "Deep-learning-based millimeter-wave massive MIMO for hybrid precoding," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 3027–3032, Mar. 2019.

[23] Y. Ma, H. He, S. Song, J. Zhang, and K. B. Letaief, "Low-complexity CSI feedback for FDD massive MIMO systems via learning to optimize," 2024. [Online]. Available: https://arxiv.org/abs/2406.16323

[24] Y. J. Noh and K. W. Choi, "Transformer-based site-specific channel estimation," in *2025 IEEE Wireless Communications and Networking Conference (WCNC)*, 2025, pp. 1–6.

[25] Z. Liu, L. Wang, L. Xu, and Z. Ding, "Deep learning for efficient CSI feedback in massive MIMO: Adapting to new environments and small datasets," *IEEE Transactions on Wireless Communications*, vol. 23, no. 9, pp. 12 297–12 312, 2024.

[26] A. Alkhateeb, G. Leus, and R. W. Heath, "Limited feedback hybrid precoding for multi-user millimeter wave systems," *IEEE Transactions on Wireless Communications*, vol. 14, no. 11, pp. 6481–6494, 2015.

[27] J. Chen, J. Tao, S. Luo, S. Li, C. Zhang, and W. Xiang, "A deep learning driven hybrid beamforming method for millimeter wave MIMO system," *Digital Communications and Networks*, vol. 9, no. 6, pp. 1291–1300, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352864822001468

[28] M. Akrout, A. Feriani, F. Bellili, A. Mezghani, and E. Hossain, "Domain generalization in machine learning models for wireless communications: Concepts, state-of-the-art, and open issues," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 4, pp. 3014–3037, 2023.

[29] J. Cheng, W. Chen, J. Xu, Y. Guo, L. Li, and B. Ai, "Swin transformer-based CSI feedback for massive MIMO," 2024. [Online]. Available: https://arxiv.org/abs/2401.06435

[30] R. Ren, J. Mo, and M. Tao, "Semcsinet: A semantic-aware CSI feedback network in massive MIMO systems," 2025. [Online]. Available: https://arxiv.org/abs/2505.08314

[31] A. Chowdhury, G. Verma, A. Swami, and S. Segarra, "Deep graph unfolding for beamforming in MU-MIMO interference networks," *IEEE Transactions on Wireless Communications*, vol. 23, no. 5, pp. 4889–4903, 2024.

[32] Y.-H. Chen, J. Emer, and V. Sze, "Using dataflow to optimize energy efficiency of deep neural network accelerators," *IEEE Micro*, vol. 37, no. 3, pp. 12–21, 2017.

[33] A. Boroumand, S. Ghose, Y. Kim, R. Ausavarungnirun, E. Shiu, R. Thakur, D. Kim, A. Kuusela, A. Knies, P. Ranganathan, and O. Mutlu, "Google workloads for consumer devices: Mitigating data movement bottlenecks," in *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 316–331. [Online]. Available: https://doi.org/10.1145/3173162.3173177

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

[35] E. Björnson, L. Sanguinetti, J. Hoydis, and M. Debbah, "Optimal design of energy-efficient multi-user MIMO systems: Is massive MIMO the answer?" *IEEE Transactions on Wireless Communications*, vol. 14, no. 6, pp. 3059–3075, 2015.

[36] S. K. Mohammed, E. G. Larsson, F. Tufvesson, and T. L. Marzetta, "Per-antenna constant envelope precoding for large multi-user MIMO systems," *IEEE Transactions on Communications*, vol. 61, no. 3, pp. 1059–1071, 2013.

[37] O. El Ayach, S. Rajagopal, S. Abu-Surra, Z. Pi, and R. W. Heath, "Spatially sparse precoding in millimeter wave MIMO systems," *IEEE Transactions on Wireless Communications*, vol. 13, no. 3, pp. 1499–1513, 2014.

[38] H. Q. Ngo, E. G. Larsson, and T. L. Marzetta, "Energy and spectral efficiency of very large multiuser MIMO systems," *IEEE Transactions on Communications*, vol. 61, no. 4, pp. 1436–1449, 2013.

[39] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, and T. Blankevoort, "A white paper on neural network quantization," *arXiv preprint arXiv:2106.08295*, 2021.

[40] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "EIE: Efficient inference engine on compressed deep neural network," in *ISCA*, 2016.

[41] S. Koppula, L. Orosa, A. G. Yağlıkçı, R. Azizi, T. Shahroodi, K. Kanellopoulos, and O. Mutlu, "EDEN: Enabling energy-efficient, high-performance deep neural network

inference using approximate DRAM," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '52. New York, NY, USA: Association for Computing Machinery, 2019, p. 166–181. [Online]. Available: https://doi.org/10.1145/3352460.3358280

[42] A. Chauhan, U. Tiwari, and N. R. Vikram, "Post training mixed precision quantization of neural networks using first-order information," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2023, pp. 1–10.

[43] M. Nagel, M. van Baalen, T. Blankevoort, and M. Welling, "Data-free quantization through weight equalization and bias correction," 2019. [Online]. Available: https://arxiv.org/abs/1906.04721

[44] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization," in *International Conference on Machine Learning (ICML)*, 2020, pp. 4479–4488.

[45] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," in *International Conference on Learning Representations (ICLR)*, 2019.

[46] Z. Dong, Z. Yao, A. Gholami, M. W. Mahoney, and K. Keutzer, "Hawq-v2: Hessian aware trace-weighted quantization of neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[47] Y. Sekikawa and S. Yashima, "Bit-pruning: A sparse multiplication-less dot-product," in *International Conference on Learning Representations (ICLR)*, 2023.

[48] D. Zhang, J. Yang, D. Ye, and G. Hua, "Lq-nets: Learned quantization for highly accurate and compact deep neural networks," 2018. [Online]. Available: https://arxiv.org/abs/1807.10029

[49] J. Choi, Z. Wang, S. Venkataramani, P. Chuang, V. Srinivasan, and K. Gopalakrishnan, "Pact: Parameterized clipping activation for quantized neural networks," in *arXiv preprint arXiv:1805.06085*, 2018.

[50] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," 2017. [Online]. Available: https://arxiv.org/abs/1608.08710

[51] K. Balaskas, A. Karatzas, C. Sad, K. Siozios, I. Anagnostopoulos, G. Zervakis, and J. Henkel, "Hardware-aware DNN compression via diverse pruning and mixed-precision quantization," *arXiv preprint arXiv:2312.15322*, 2023.

[52] C. Liu, R. Zhang, X. Zhang, Y. Hao, Z. Du, X. Hu, L. Li, and Q. Guo, "Ultra-low precision multiplication-free training for deep neural networks," *arXiv preprint arXiv:2302.14458*, 2023.

[53] F. Karimzadeh, J. H. Yoon, and A. Raychowdhury, "Bits-net: Bit-sparse deep neural network for energy-efficient RRAM-based compute-in-memory," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 5, pp. 1952–1961, 2022.

[54] S. I. Venieris, A. Kouris, and C.-S. Bouganis, "Toolflows for mapping convolutional neural networks on fpgas: A survey and future directions," *ACM Comput. Surv.*, vol. 51, no. 3, Jun. 2018. [Online]. Available: https://doi.org/10.1145/3186332

[55] L. Zhao, K. Shao, F. Tian, T. K.-T. Cheng, C.-Y. Tsui, and Y. Zou, "A flexible precision scaling deep neural network accelerator with efficient weight combination," in *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2025, pp. 1–5.

[56] Y. Umuroglu, N. J. Fraser, G. Gambardella, M. Blott, P. Leong, M. Jahre, and K. Vissers, "Finn: A framework for fast, scalable binarized neural network inference," in *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, ser. FPGA '17. ACM, Feb. 2017, p. 65–74. [Online]. Available: http://dx.doi.org/10.1145/3020078.3021744

[57] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2017.

[58] B. Wu, X. Dai, P. Zhang, Y. Wang, F. Wu, and K. Keutzer, "Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 10 734–10 742.

[59] H. Cai, L. Zhu, and S. Han, "Proxylessnas: Direct neural architecture search on target task and hardware," in *International Conference on Learning Representations (ICLR)*, 2019.

[60] X. Zhang, H. Zhao, H. Zhu, B. Adebisi, G. Gui, H. Gacanin, and F. Adachi, "Nas-amr: Neural architecture search-based automatic modulation recognition for integrated sensing and communication systems," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 3, pp. 1374–1386, 2022.

[61] A. Alkhateeb, J. Mo, N. Gonzalez-Prelcic, and R. W. Heath, "MIMO precoding and combining solutions for millimeter-wave systems," *IEEE Communications Magazine*, vol. 52, no. 12, pp. 122–131, 2014.

[62] 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. 37 328–37 348, 2018.

[63] A. H. Karkan, H. Hojatian, J.-F. Frigon, and F. Leduc-Primeau, "SAGE-HB: Swift adaptation and generalization in massive MIMO hybrid beamforming," in *2024 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*, 2024, pp. 323–328.

[64] A. H. Karkan, A. Ibrahim, J.-F. Frigon, and F. Leduc-Primeau, "A low-complexity plug-and-play deep learning model for massive MIMO precoding across sites," *IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*, 2025.

[65] A. M. Elbir, A. Papazafeiropoulos, Z. Ding, and E. Björnson, "Federated learning for hybrid beamforming in mmwave massive MIMO systems," *IEEE Transactions on Wireless Communications*, vol. 20, no. 12, pp. 8320–8334, 2021.

[66] J. Choi, J. Park, and N. Lee, "Energy efficiency maximization precoding for quantized massive MIMO systems," *IEEE Transactions on Wireless Communications*, vol. 21, no. 9, pp. 6803–6817, 2022.

[67] S. Tridgell, D. Boland, P. H. Leong, R. Kastner, A. Khodamoradi, and Siddhartha, "Real-time automatic modulation classification using RFSoC," in *2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2020, pp. 82–89.

[68] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE transactions on information theory*, vol. 44, no. 6, pp. 2325–2383, 2002.

[69] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," in *4th Int. Conf. on Learning Representations, (ICLR)*, 2016.

[70] J. Choi, Z. Wang, S. Venkataramani, P. I. Chuang, V. Srinivasan, and K. Gopalakrishnan, "PACT: parameterized clipping activation for quantized neural networks," 2018. [Online]. Available: http://arxiv.org/abs/1805.06085

[71] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *Journal of Machine Learning Research*, vol. 20, no. 55, pp. 1–21, 2019.

[72] H. Hojatian, Z. Mlika, J. Nadal, J.-F. Frigon, and F. Leduc-Primeau, "Learning energy-efficient transmitter configurations for massive MIMO beamforming," *IEEE Trans. on Machine Learning in Communications and Networking*, 2024.

[73] R. Desislavov, F. Martínez-Plumed, and J. Hernández-Orallo, "Trends in AI inference energy consumption: Beyond the performance-vs-parameter laws of deep learning," *Sustainable Computing: Informatics and Systems*, vol. 38, p. 100857, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210537923000124

[74] Y. Li, W. Wang, H. Bai, R. Gong, X. Dong, and F. Yu, "Efficient bitwidth search for practical mixed precision neural network," 2020. [Online]. Available: https://arxiv.org/abs/2003.07577

[75] B. Moons, K. Goetschalckx, N. Van Berckelaer, and M. Verhelst, "Minimum energy quantized neural networks," in *2017 51st Asilomar Conf. on Signals, Systems, and Computers*, Oct 2017, pp. 1921–1925.

[76] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org, 2017.

[77] J. Emery, A. H. Karkan, J.-F. Frigon, and F. Leduc-Primeau, "A foundation model for massive MIMO precoding with an adaptive per-user rate-power tradeoff," in *2025 IEEE 36th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2025, to appear.