Joint High-Dimensional Soft Bit Estimation and Quantization using Deep Learning

Marius Arvinte (✉ arvinte@utexas.edu)

The University of Texas at Austin <https://orcid.org/0000-0002-9328-9223>

Sriram Vishwanath

The University of Texas at Austin

Ahmed H. Tewfik

The University of Texas at Austin

Jonathan I. Tamir

The University of Texas at Austin <https://orcid.org/0000-0001-9113-9566>

Research

Keywords: Soft Bits, Deep Learning, Quantization, Estimation

Posted Date: February 3rd, 2022

DOI: <https://doi.org/10.21203/rs.3.rs-1304406/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Joint High-Dimensional Soft Bit Estimation and Quantization using Deep Learning

Marius Arvinte^{*}, Sriram Vishwanath, Ahmed H. Tewfik and Jonathan I. Tamir^{*}

^{*}Correspondence:

arvinte@utexas.edu;

jtamir@utexas.edu

Department of Electrical and
Computer Engineering, University
of Texas at Austin, Austin, USA

Full list of author information is
available at the end of the article

Abstract

Forward error correction using soft probability estimates is a central component in modern digital communication receivers and impacts end-to-end system performance. In this work, we introduce EQ-Net: a deep learning approach for joint soft bit estimation and quantization in high-dimensional multiple-input multiple-output (MIMO) systems. We propose a two-stage algorithm that uses soft bit quantization as pre-training for estimation, and is motivated by a theoretical analysis of soft bit compression bounds in MIMO channels. Our experiments demonstrate that a single deep learning model achieves competitive results on both tasks when compared to previous methods, with gains in quantization efficiency as high as 20% and reduced estimation latency by at least 21% compared to other deep learning approaches that achieve the same end-to-end performance. We investigate the robustness of the proposed approach, and demonstrate that the model is robust to distributional shifts when used for quantization, and is competitive with state-of-the-art deep learning approaches when faced with channel estimation errors in the task of soft bit estimation.

Keywords: Soft Bits; Deep Learning; Quantization; Estimation

1 Introduction

Digital MIMO systems operate by transmitting a vector of discrete symbols across a single channel use and are a major component of current 5G and envisioned 6G communication systems [1]. Given the prevalent use of error-correcting codes, such as low-density parity check (LDPC) codes, two core tasks for modems in these systems consist in soft bit estimation (also commonly referred to as soft-output MIMO detection, or soft detection) and quantization. Since utilizing soft instead of hard bit estimates for decoding error-correcting codes is known to provide an order

of magnitude order in reduction of end-to-end error rates [2], algorithms that operate on soft bits must ensure that they are close as possible to the optimal solution. In this work, our goal is to develop learning-aided solutions for two different tasks, handled under the same framework: soft bit estimation and quantization. Even though the two tasks require different inputs, we develop a single deep learning framework that addresses them simultaneously and exploits the information learned during quantization for better supervised training of an estimation algorithm.

Soft bit estimation is a challenging practical problem due to the exponential complexity of the optimal solution [3] and stringent latency requirements in 5G-and-beyond communication systems [4]. For example, given a massive MIMO 5G system operating at sub-6 GHz, the base station could be faced with estimating soft bits from as many as thousands of channel uses per data frame [5], each of these requiring an expensive algorithm if optimal or near-optimal performance is desired. Near-optimal estimation algorithms are a central part of end-to-end performance in coded systems (e.g., low-density parity-check (LDPC) codes) [6], with solutions that use machine learning to obtain high-performance and low-latency being an active area of research. Our work builds upon this line of research, and introduces a deep learning aided algorithm for near-optimal soft bit estimation.

In parallel, soft bit quantization is required for storing a large number of soft bits for a potentially long duration of time, such as in hybrid automatic repeat request (HARQ) schemes [7] or relay channels [8]. Soft bit quantization is also important in systems where energy and memory efficiency represent practical limitations. For example, in distributed communications systems [9], low-resolution soft bit quantization is required whenever relaying or feedback is involved, since system capacity represents a bottleneck [10], and soft bits must be transmitted without errors under a low communication budget. Quantization is also required in systems that use the HARQ protocol, where it is beneficial for the receiver to store soft bit values from a failed transmission and use a soft combining scheme [11] to boost performance, making storage a potential system bottleneck.

To address these issues, deep learning methods have emerged as promising candidates for aiding or completely replacing signal processing blocks in MIMO communication systems [12,13]. In this work, we focus on the case where deep learning is used on specific functional tasks in end-to-end communication systems. These approaches

are attractive since they are modular and compatible with already deployed implementations of communication protocols. While previous solutions have been developed for both the task of soft bit estimation and quantization, there is currently no solution that addresses both tasks simultaneously. Furthermore, the robustness of deep learning based methods is still an open problem in the broader machine learning field [14] and has been recognized as an issue in digital communications as well [15, 16]. In this paper, we address this research gap and introduce EQ-Net, a data-driven architecture that aims to solve the challenges of low-latency quantization and estimation, while still retaining the end-to-end system performance of optimal approaches and outperforming competing deep learning approaches.

1.1 Related Work

1.1.1 Soft Bit Estimation

For the task of soft bit estimation, there are a broad number of classical (non-learning based) signal processing algorithms that have been developed, given that channel decoding using soft bit inputs is ubiquitous in practice [2]. A modern survey of MIMO soft bit estimation (also called soft-output detection) is presented in [6]. The V-BLAST algorithm [17] uses the idea of sequential estimation, with subsequent work leading to zero-forcing with successive interference cancellation (ZF-SIC) [18] as an algorithm for efficient detection. In this method, the system is reduced to its upper triangular form and data symbols are detected in a fixed order. Once a symbol is detected, it is assumed to be correct and subtracted from the remaining data streams. This leads to a low-complexity, but also low-performance method. Sphere decoding [19, 20] formulates the detection problem as a tree search algorithm and performs a greedy search. In the soft-output version [20], multiple candidate solutions are used to estimate log-likelihoods. A drawback of this class of algorithms is the need for specialized hardware to fit into tight latency budgets in 5G and beyond scenarios [21]. To address latency, recent work combines sphere decoding with deep learning for search radius and branch prediction [22].

More recently, there has been work in model-based approaches for MIMO detection, where optimization steps (e.g., application of the gradient descent algorithm) are combined with learnable blocks via deep learning. The work in [23] proposes OAMP-Net2 as a data-based extension to the orthogonal approximate

message passing (OAMP) detection algorithm [24], where step sizes are treated as learnable parameters. This method has the advantage of a very small number of learnable parameters, but has a relatively large end-to-end latency because of the matrix operations involved during inference. The work in [25] takes a similar approach, but replaces the fixed computations of the OAMP algorithm with fully learnable transforms (i.e., layers of a deep neural network), resulting in competitive results for MIMO soft bit estimation and an architecture that can be scaled to high-dimensional scenarios. Finally, the work in [26] trains a two-layer deep neural network using a supervised loss directly on the soft bits, in single-input single-output (SISO) channels, leveraging that, in this case, a closed-form linear approximation of the soft bits exists. This is not the case for MIMO detection, where single-stage supervised training may encounter convergence issues, as outlined in our experiments.

1.1.2 Soft Bit Quantization

Generally, there are two approaches developed in the prior work regarding soft bit quantization: scalar or vector quantization. In the scalar approach, each soft bit is separately quantized, independent of the others, and without considering structure at a channel use level. The work in [27] introduces an information-theoretic optimal data-based approach for quantizing soft bits from arbitrary distributions, such as corresponding to the same bit position in Gray-coded digital quadrature amplitude modulation (QAM). This approach also has the advantage that it does not make any assumptions about the underlying channel model and an algorithm is given for estimating optimal scalar quantization levels in arbitrary channels. The work in [28] proposes a solution for soft bit quantization in relay systems based on maximizing the mutual information between two transmitters and one receiver. While both of these approaches are competitive, they do not take advantage of redundancy between soft bit estimates derived from the same channel use.

A promising research avenue is to consider vector quantization techniques for soft bits in MIMO channels. The work in [29] extends the data-driven approach in [30] and proposes a vector extension to a maximum mutual information quantizer, but loses theoretical optimality guarantees. The method in [31] introduces a deep learning approach that leverages the redundancy between soft bit values corresponding

to a single channel use and achieves excellent quantization results for SISO channels. However, there remains the issue of developing a vector quantization method for arbitrary MIMO scenarios, which is a major distinction between this paper and all other prior work in terms of quantization methods. Also, prior work does not discuss or exploit in any way the learned representation beyond the quantization task (i.e., for estimation), which is also a central part of EQ-Net.

1.2 Contributions

Our contributions in this work are the following:

- 1 We prove lower and upper compression bounds (Theorems 1 and 2) on the maximum likelihood estimate of the soft bits in arbitrary MIMO channels. The upper bound is proved constructively through a construction that holds for any channel, while the lower bound comes from the class of diagonalized MIMO channels. In practice, we verify that the lower bound can be achieved *for arbitrary channels* and learned using deep neural networks, which becomes a key design criterion in the proposed approach.
- 2 We introduce a methodology for the supervised training of a joint soft bit estimation (E) and quantization (Q) algorithm in MIMO scenarios, termed EQ-Net. The proposed approach involves a two-stage training procedure: the first stage trains a deep autoencoder for quantization, while the second stage trains an estimation encoder while re-using the quantization decoder. We experimentally verify that the two-stage training algorithm improves convergence and has better estimation performance than single-stage supervised training algorithms, when the deep neural networks are constrained to a fixed depth.
- 3 We experimentally evaluate end-to-end block error rate performance, inference latency and the robustness of EQ-Net in soft bit quantization and estimation. We demonstrate competitive results in both tasks simultaneously, under realistic system simulations. We compare our method with classical signal processing algorithms, as well as deep learning-based approaches. We obtain gains of up to 20% in numerical quantization efficiency and estimation gains of up to 1 dB in end-to-end system performance in coded MIMO orthogonal frequency-division multiplexing (OFDM) scenarios and demonstrate latency

improvements against other deep learning approaches. Finally, we demonstrate that EQ-Net is robust to inaccurate channel state information (CSI) and different train-test distributions of the channel conditions when evaluated on the quantization task, and is competitive with other deep learning methods on the robust estimation task.

2 Methods

2.1 System Model

We assume a narrowband, instantaneous, digital MIMO communication model. This encompasses practical scenarios, such as single carrier MIMO communication or an individual subcarrier of a MIMO-OFDM transmission, and is flexible enough to model various distributions of the MIMO channel matrix. The communication model is given by [32, Eq. 7.55]:

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}, \quad (1)$$

where $\mathbf{x} \in \mathbb{C}^{N_t}$ is a vector of transmitted symbols drawn from a finite, discrete constellation $\mathcal{C}$, and $\mathbf{y} \in \mathbb{C}^{N_r}$ is a vector of received symbols. N_t and N_r represent the number of transmitted and received data streams, respectively, and $\mathbf{n} \in \mathbb{C}^{N_r}$ is an i.i.d. complex Gaussian noise vector with covariance matrix $\sigma^2 \mathbf{I}$. For the remainder of this work, we deal with spatial multiplexing scenarios, where $N_r \geq N_t$ and the transmitted data streams are assumed to be independent, but an application of our method to transmit diversity is possible when considering the effective digital matrix as the product between the precoding matrix and the channel matrix. $\mathbf{H}$ is the digital channel matrix characterizing the baseband channel effects between the transmitter and the receiver, and includes the effects of precoding and beamforming. In practice, $\mathcal{C}$ is the set of symbols in a QAM constellation with 2^K elements, where K is the modulation order and represents the number of information bits transmitted in each of the N_t data streams.

The k -th log-likelihood ratio of the i -th transmitted symbol is defined as [27]:

$$L_{i,k} = \log \frac{P(\mathbf{y}|b_{i,k} = 1)}{P(\mathbf{y}|b_{i,k} = 0)}. \quad (2)$$

For the remainder of this work, we use the notion of *soft bits*, which are closely related to the log-likelihood ratios through an invertible transform [27] and are used

in practical decoders for error-correcting codes due to operations being simpler to carry out in the hyperbolic tangent domain [33]:

$$\Lambda_{i,k} = \tanh \frac{L_{i,k}}{2}. \quad (3)$$

In scenarios where the noise is non-zero, soft bits are constrained to the interval $(-1, 1)$, with values asymptotically close to 1 and -1 leading to near-certain probabilities that the corresponding bit position is equal to one or zero, respectively. This is beneficial for data-driven methods, since a limited dynamic range is also helpful in the stability of deep learning methods, where unbounded inputs can lead to exploding gradients [34]. By replacing the definition of the log-likelihood ratio from (2), we obtain that the soft bits are defined as:

$$\Lambda_{i,k} = \frac{P(\mathbf{y}|b_{i,k} = 1) - P(\mathbf{y}|b_{i,k} = 0)}{P(\mathbf{y}|b_{i,k} = 1) + P(\mathbf{y}|b_{i,k} = 0)}. \quad (4)$$

We organize the soft bits corresponding to a single MIMO channel use in the soft bit matrix $\mathbf{\Lambda} \in \mathbb{R}^{N_t \times K}$. The likelihoods $P(\mathbf{y}|b_{i,k})$ in (4) can be further expanded by using the total probability law. For the soft bit corresponding to the i -th transmitted stream and the k -th bit position, we let $\bar{b}_{i,k} = \{b_{j,l} | (j, l) \neq (i, k)\}$ be the set of all remaining bit positions. Then, the conditional probability of a particular bit position can be expanded via marginalization as:

$$P(\mathbf{y}|b_{i,k}) = \sum_{\bar{b}_{i,k} \in \{0,1\}^{N_t K - 1}} P(\mathbf{y}, \bar{b}_{i,k} | b_{i,k}) \quad (5)$$

where the sum is taken over all possible values of the vector $\bar{b}_{i,k}$, i.e., all bits except the position of interest are marginalized. Since the terms in (4) involve conditioning the bit $b_{i,k}$ on a specific outcome and together with the marginalization in (5), it follows that, under the i.i.d. Gaussian noise assumption, each probability can be expanded as (up to a normalization factor):

$$P(\mathbf{y}, \bar{b}_{i,k} | b_{i,k}) \propto \sum_{\mathbf{x} \in \mathcal{C} | \mathbf{x} = \text{QAM}(\bar{b}_{i,k}, b_{i,k})} \exp \left(-\frac{\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2}{\sigma^2} \right), \quad (6)$$

where we use $\mathbf{x}$ to denote each symbol corresponding to a specific combination of $\bar{b}_{i,k}, b_{i,k}$ obtained by using the given QAM constellation. Each exponential in the sum corresponds to the conditional probability of the observed symbols, assuming

a specific vector was transmitted and comes from the Gaussian noise assumption. Finally, the optimal maximum-likelihood (ML) estimator for the soft bits in channels affected by i.i.d. Gaussian noise can be expressed as:

$$\Lambda_{i,k}^{(\text{ML})} = \frac{\sum_{\mathbf{x} \in \mathcal{C} | b_{i,k}=1} \exp\left(-\frac{\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2}{\sigma^2}\right) + \sum_{\mathbf{x} \in \mathcal{C} | b_{i,k}=0} \exp\left(-\frac{\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2}{\sigma^2}\right)}{\sum_{\mathbf{x} \in \mathcal{C} | b_{i,k}=1} \exp\left(-\frac{\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2}{\sigma^2}\right) - \sum_{\mathbf{x} \in \mathcal{C} | b_{i,k}=0} \exp\left(-\frac{\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2}{\sigma^2}\right)}. \quad (7)$$

The goal of soft bit estimation is to design a function f_E that takes as input the received symbols $\mathbf{y}$, channel $\mathbf{H}$, the symbol constellation $\mathcal{C}$ and the noise standard deviation σ and outputs an estimate of the soft bit matrix $\tilde{\mathbf{\Lambda}}$. While (7) is the optimal estimator for the soft bits when the noise is Gaussian and independent – a common situation in practice [35] – the two distinct sums in (7) each involve a number of $2^{N_t K - 1}$ terms. This leads to a prohibitive computational complexity even for moderate values of N_t and K , and algorithms that approximate the solution $\mathbf{\Lambda}^{(\text{ML})}$ by reducing the amount of performed computations are the main goal in efficient soft bit estimation.

The goal of soft bit quantization is to design the following pair of functions, given the full precision ML soft bit estimate $\mathbf{\Lambda}^{(\text{ML})}$:

- An encoder f_Q that compresses the matrix $\mathbf{\Lambda}^{(\text{ML})}$ to a lower or equal dimension representation $\mathbf{z}$ and numerically quantizes $\mathbf{z}$ to a bit stream of finite length. In general, this function is lossy, and the bit stream encodes an entry in a quantization codebook that is fixed and is known to both the transmitter and receiver (e.g., similar to how beamforming codebooks are standardized in 5G systems [35]).
- A decoder g_Q that recovers $\mathbf{\Lambda}^{(\text{ML})}$ as accurately as possible from a given bit stream, with respect to a pre-defined error function.

We highlight that, while the tasks of soft bit quantization and estimation are different in terms of what inputs are available during deployment (inference), they share a common characteristic: for both tasks, the output has to approximate the optimal ML solution as closely as possible. We use the following weighted mean squared error loss to quantify this deviation, which represents an extension of the

loss in [31] to MIMO scenarios:

$$\mathcal{L}(\tilde{\mathbf{\Lambda}}, \mathbf{\Lambda}^{(\text{ML})}) = \sum_{i=1}^{N_t} \sum_{k=1}^K \frac{(\tilde{\Lambda}_{i,k} - \Lambda_{i,k}^{(\text{ML})})^2}{|\Lambda_{i,k}^{(\text{ML})}| + \epsilon}, \quad (8)$$

where $\tilde{\mathbf{\Lambda}}$ is the output of the estimator. $\mathbf{\Lambda}^{(\text{ML})}$ is the desired output (in this case, the ML estimate), and the term ϵ acts as a stabilization constant, since some bits can have an arbitrarily small absolute value, and would thus need to be reconstructed or estimated to an arbitrary precision. Since the impact of the soft bits with very low magnitude values is negligible in decoding algorithms [33], we use a value of $\epsilon = 0.001$ to bound the loss in (8), chosen as a trade-off between stability and accuracy.

2.2 Theoretical Derivations

In this section, we prove a lower and upper bound on the compression ratio of the soft bit matrix $\mathbf{\Lambda}^{(\text{ML})}$. We define the compression ratio of a surjective mapping $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ as the ratio m/n , where any function with a ratio strictly higher than one can be used to represent the data (co-domain of f) using a lower-dimensional feature space, thus achieve compression. Note that the theory in this section does not cover numerical quantization and requires infinite numerical precision for the domain and co-domain of f . In the experimental result section, numerical quantization is applied to this feature space to achieve the final soft bit quantization algorithm, and its impact is evaluated.

Even assuming infinite numerical precision, the two bounds derived in this section are helpful in choosing the feature dimension in a learning-based compression architecture, where we would ideally like to always use as *few* features as possible, and further quantize each of them using a fixed bit budget. We prove that the lower bound is achievable for arbitrary channels by constructing a feature representation that achieves it. For the upper bound, we prove that it holds and can be achieved for the special class of diagonalized channels, and in practice we attempt and are successful in learning a deep compression scheme that achieves the upper bound for arbitrary channels, even under test-time distributional shifts.

2.2.1 Lower Bound: Compressibility of the ML Estimate in Arbitrary Channels

We directly work with (1) and make no assumptions on $\mathbf{x}$, other that it is sampled from the constellation $\mathcal{C}$. We prove the following lower bound for the compression ratio.

Theorem 1 *Let $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$ be the noisy received symbols in arbitrary channels. Then, there exists a surjective function $f : \mathbb{R}^{N_t(N_t+2)} \rightarrow \mathbb{R}^{K \times N_t}$ such that $f(\mathbf{z}) = \mathbf{\Lambda}^{(ML)}$ and a compression ratio of $R_{\text{low}} = K/(N_t + 2)$ is achieved.*

The proof is provided in the Appendix and relies on the QR decomposition of $\mathbf{H}$, and recognizing that after equalizing the $\mathbf{Q}$ term, the statistics of the noise do not change, and the system is now represented by an upper triangular equation. The representation of the upper triangular channel itself takes $\mathcal{O}(N_t^2)$ degrees of freedom, because we do not make any structural assumptions about the matrix $\mathbf{H}$, and storing the raw channel tensor and the received symbols – which take $\mathcal{O}(N_t)$ degrees of freedom – is sufficient for exact soft bit recovery using a modified form of (4).

2.2.2 Upper Bound: Compressibility of the ML Estimate in Diagonalized Channels

We consider a version of (1) where the transmitter has knowledge of the channel matrix $\mathbf{H}$. Furthermore, we assume that the transmitted symbols take the form $\mathbf{s} = \mathbf{V}\mathbf{x}$, where $\mathbf{V}$ represents the matrix of right singular vectors of $\mathbf{H}$ and $\mathbf{x}$ is drawn from $\mathcal{C}$. This type of linear precoding is shown to be optimal when using water-filling [36] under transmit power constraints. Then, the following theorem holds.

Theorem 2 *Let $\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n}$ be the noisy received symbols, where $\mathbf{s} = \mathbf{V}\mathbf{x}$ are the transmitted symbols. Then, there exists a surjective function $f : \mathbb{R}^{3N_t} \rightarrow \mathbb{R}^{K \times N_t}$ such that $f(\mathbf{z}) = \mathbf{\Lambda}^{(ML)}$ and a compression ratio of $R_{\text{up}} = K/3$ is achieved.*

The proof for the upper bound is provided in the Appendix. Intuitively, this is done by recognizing that the channel becomes diagonal after performing left-side multiplication with the conjugate transpose of the matrix of left singular vectors $\mathbf{U}$. Then, since the MIMO channel can be separated in a set of N_t virtual channels, the soft bits corresponding to each virtual channel can be exactly represented using

a three dimensional vector, regardless of K [31]. Note that the term compression here refers to lossless representations achieved in infinite numerical precision and does not involve numerical quantization (hence the domain of f being $\mathbb{R}^{3N_t}$). The impact of finite precision quantization is discussed and evaluated in Section 3.

In practice, the lower bound of $K/(N_t + 2)$ is too weak to use for compression, since in many cases it is possible that $N_t \geq K - 2$, and thus no practical compression can be achieved. On the other hand, the upper bound is *invariant* with respect to the number of transmitted data streams N_t and always compresses for $K \geq 4$ (i.e., a modulation of 16-QAM or higher order). Note that the two bounds coincide and a feature representation that achieves them can be derived in closed-form for the particular case of $N_t = 1$ (SISO channels), which recovers the previous work in [31]. However, achieving the upper bound is non-trivial in arbitrary channels.

A key component of EQ-Net is to assume that the upper bound can be achieved in arbitrary – and not only in diagonalized – MIMO channels. This implies that a feature space of size $3N_t$ is sufficient to represent the matrix of soft bits in any MIMO channel. While we do not derive such a representation explicitly, this motivates us to attempt and learn it by using the representational power of deep neural networks. In particular, this makes a deep autoencoder a very suitable choice, since we can control the dimension of the bottleneck feature representation and set it to $3N_t$, which is a core idea behind the EQ-Net approach.

2.3 The Proposed EQ-Net Architecture: Joint Estimation and Quantization

The previous section motivates EQ-Net as a feature learning algorithm with a compressed feature representation from which the soft bit matrix is recovered. The size of this feature space is always equal to $3N_t$ real-valued features and we would like to robustly map soft bits to and recover them from this representation in arbitrary MIMO channels, with as little added distortions as possible between the output of the autoencoder and the ML estimate. We present a data-driven approach for learning a model which achieves near-ML end-to-end performance.

EQ-Net is a supervised method that uses a deep autoencoder and leverages compression as a supervised pre-training task for learning a soft bit estimator: when training the estimator, we do not learn a direct mapping from the received symbols to the soft bit matrix, but rather split the learning in two separate stages:

- 1 In the first stage, we train a pair of quantization encoder and decoder functions by compressing to a feature representation of size $3N_t$ (the upper bound in Theorem 2, but without any assumptions about the channel structure). To enable end-to-end learning, we approximate the quantization function for the forward and backward passes.
- 2 In the second stage, once the quantization autoencoder has been trained, we train an additional estimation encoder that takes the received symbols and CSI, and is trained to predict the features corresponding to the soft bits obtained from the first stage. Importantly, in this stage, the features are frozen and only this new encoder is learned.

A high-level functional diagram of EQ-Net is shown in Figure 1. Our ablation experiments in Section 3 show that the two-stage procedure – where we first train the pair f_Q and g_Q , and then re-use them to learn f_E – is an essential step when training a model with limited depth and width and that single stage end-to-end training falls into unwanted local minima.

Stage 1: Compression and Quantization In the first stage, we train an autoencoder consisting of a quantization encoder f_Q and a decoder g_Q . During both training and testing, the input to the quantization encoder is the ML estimate of the soft bits $\Lambda^{(\text{ML})}$ or, when infeasible due to the large complexity, the estimate of an approximate algorithm such as soft-output sphere detection. The parameters of the quantization encoder and decoder are denoted by θ_f and θ_g . Let the latent feature representation output by the encoder be:

$$\mathbf{z} = f_Q \left(\Lambda^{(\text{ML})}; \theta_f \right). \quad (9)$$

We include $\mathbf{z}$ as a separate functional block in Figure 1 since this feature representation is essential for the second stage. Additionally, let $\tilde{\Lambda}$ denote the matrix of reconstructed soft bits:

$$\tilde{\Lambda} = g_Q \left(\mathcal{Q}(\mathbf{z}); \theta_g \right), \quad (10)$$

where $\mathcal{Q}$ is a function that maps the interval $[-1, 1]$ to a discrete and finite set of points $\mathcal{C}$ of size 2^{N_b} , where N_b is the length of the quantized bit stream. During training, we replace this operator with a differentiable approximation to obtain

useful gradients by following previous work [31] and using the noise model $\mathcal{Q}(x) = x + u$, where u is drawn from $\mathcal{N}(0, \sigma_u)$ and $\sigma_u = 0.001$ is chosen small enough to avoid an error floor in full precision. To prevent the autoencoder from learning a solution that amplifies the magnitude of the latent feature representation and decreases the relative power of the added noise, we use a hyperbolic tangent activation on the output layer of f_Q to strictly bound the magnitude of $\mathbf{z}$.

The loss function used to learn the parameters θ_f and θ_g of the quantization autoencoder is given by:

$$\mathcal{L}_Q(\theta_f, \theta_g) = \frac{1}{B} \sum_{n=1}^B \mathcal{L}(\tilde{\mathbf{\Lambda}}^{(n)}, \mathbf{\Lambda}^{(n, \text{ML})}), \quad (11)$$

where B is the batch size and $\mathcal{L}$ is given by (8), applied element-wise and summed. Once the quantization autoencoder is trained we obtain the feature representation $\mathbf{z}$ for all training samples by performing a forward pass with the encoder f_Q .

Stage 2: Estimation In the second stage, we train an estimation encoder f_E with parameters θ_e . This model uses the received data streams $\mathbf{y}$ and available CSI to recover the feature representation learned by the quantization encoder. Given a pretrained pair of quantization functions, we can then further use the quantization decoder to recover the estimated soft bits. The supervised loss used to train the parameters θ_E is given by:

$$\mathcal{L}_E(\theta_e) = \frac{1}{B} \sum_{n=1}^B \left\| f_E(\mathbf{y}^{(n)}, \mathbf{H}^{(n)}, \sigma^{(n)}; \theta_e) - \mathbf{z}^{(n)} \right\|_1. \quad (12)$$

Note that the ground truth soft bit matrix $\mathbf{\Lambda}^{(\text{ML})}$ is never explicitly used as a target output in the second stage, and we completely rely on the pre-trained feature extractor given by f_Q . Obtaining the estimated soft bits from the estimated feature representation is straightforward during inference and is done as:

$$\tilde{\mathbf{\Lambda}} = g_Q(f_E(\mathbf{y}, \mathbf{H}, \sigma)). \quad (13)$$

Intuitively, the models f_Q and f_E are a teacher-student pair [37] when it comes to learning the feature representation of the soft bit matrix, since the features learned for the task of quantization are reused and serve as targets for training the estimation encoder. The parameters θ_f are frozen in this stage, hence not included

in the left-hand side of (12). This approach has specific advantages over single-stage supervised learning estimation approaches that are trained end-to-end to directly output the soft bits:

- Training using the two-stage approach is more stable, converges faster and to a lower weighted MSE value when we evaluate using g_Q . This claim is verified in our experiments.
- Two-stage training allows for the use of a compact and shallow estimation encoder, which benefits end-to-end latency when compared to other deep learning approaches.

At test time, EQ-Net inference is executed independently for each channel use. If the system is based on MIMO-OFDM, then multiple subcarriers can be executed in parallel for increased efficiency. This follows the lines of prior work in deep learning-based detection schemes [23, 25] and allows for compact models that can be efficiently executed using vector arithmetic units. Given the vector of received data streams $\mathbf{y}$, the channel $\mathbf{H}$ and estimated noise variance σ^2 , the architecture uses them to operate in one of the following modes.

In *quantization* mode the algorithm is compatible with any external module that estimates the soft bits. If an algorithm estimates the log-likelihood ratios, then we simply apply (3) as a pre-processing step to convert these to soft bits. The matrix of values $\mathbf{\Lambda}^{(\text{ML})}$ corresponding to a channel use is serialized and fed to the encoder f_Q , and the result is quantized with the operator $\mathcal{Q}$, yielding a bit string representing the quantized soft bit matrix. This compact bit string is stored for future use in the case of HARQ systems, or efficiently transmitted over a relay channel, after which the recovered soft bit matrix $\hat{\mathbf{A}}$ is output by applying g_Q .

In *estimation* mode the algorithm directly passes $\mathbf{y}$, $\mathbf{H}$ and σ through the estimation encoder and the decoder without quantizing the latent representation, resulting in the estimated soft bit matrix $\tilde{\mathbf{A}}$. In practice, $\mathbf{H}$ and σ come from an external channel estimation module. A hybrid mode of operation is possible, where joint quantization and estimation is performed, which is useful in applications where f_E and g_Q are distributed across different entities, such as compress-and-forward schemes [8]. In our experiments, we only evaluate the performance of EQ-Net on each task independently.

2.4 Implementation Details

The models f_Q , f_E and g_Q consist of deep neural networks that use fully-connected layers, sharing the common design principle of reducing the end-to-end latency as much as possible to make them suitable for fast inference. The most straightforward way of achieving this is by limiting the depth of all networks while potentially increasing their width. Conversely, we investigate increasing depth at the cost of latency to improve system performance in the sequel. A detailed block diagram of the models is shown in Figure 2. All models use the rectified linear unit activation function in the hidden layers, given by $\text{relu}(x) = \max\{x, 0\}$, with the few exceptions shown in Figure 2, where a hyperbolic tangent activation is used to limit the dynamic range of the learned representations. The quantization encoder f_Q uses a simple feed-forward, fully-connected deep neural network with six hidden layers, a width of $4N_t K$ for the hidden layers, and $3N_t$ for the output layer, which matches the upper bound in Theorem 2. Importantly, the width of each layer scales with modulation order, while the depth remains fixed, which is a design choice made to increase the expressive power of the network without sacrificing latency in higher-order modulation scenarios, since in our experiments the width is low enough to accommodate matrix multiplication without batching.

The quantization decoder g_Q uses the branched architecture described in [31]: the latent representation is separately processed using KN_t parallel, feed-forward, fully-connected deep neural networks, each with six hidden layers and with the same width as the encoder.

To numerically quantize the latent feature representation during testing, where gradients are no longer needed, we learn a factorized quantization codebook by separately applying a scalar quantization function to each of the latent space components. We use the same data-based approach as [31] and train a k-means++ [38] scalar quantizer after f_Q and g_Q have been trained. We learn a separate quantizer for each latent feature, since training a vector quantizer in the feature space is too expensive and scales exponentially with N_t , given that a vector codebook will have at least 2^{9N_t} entries, when allocating at least three bits for each of the $3N_t$ features. The memory cost for storing all the scalar quantization codebooks is less than 2 kB in all the experiments, and scales linearly with N_t .

The estimation encoder f_E takes as input the triplet $(\mathbf{y}, \mathbf{H}, \sigma)$ and passes their flattened, real-valued (obtained by concatenating the real and imaginary parts) versions through a separate dense layer for each signal, followed by a concatenation operation. This is an early feature fusion strategy we have used due to the three input signals having different physical dimensions and scales. A single block of the estimation encoder contains an additional six hidden layers with residual connections between them, as shown in Figure 2(c). Note that we do not normalize the data for either of the two encoders.

For training EQ-Net, we use the Adam optimizer [39] with a batch size of 32768 samples, learning rate of 0.001 and default Tensorflow [40] parameters for both stages. Training, validation and test data consists of pre-generated ML soft bit estimates using (4) from 10000 half-rate, LDPC-coded, packets at six logarithmically spaced SNR values, such that the block error rate falls in the region of interest between 1 and 10^{-4} for each scenario, and split using a 80/10/10 train/validation/test split. We focus on two system configurations: 2-by-2 MIMO with 64-QAM modulation and 4-by-4 MIMO with 16-QAM modulation. Complete hyper-parameters are available in the online code repository^[1].

3 Results and Discussion

3.1 Verifying Theorems 1 and 2

To verify the compression bounds for soft bits in MIMO channels, and whether the upper bound can be achieved, we treat the latent feature dimension as a hyper-parameter and perform an ablation experiment over its size. We train a series of quantization models (the pair of models f_Q and g_Q) where the only parameter that changes is the size of this feature representation. For exemplification, we target a 2-by-2, 64-QAM scenario under i.i.d. Rayleigh fading, but this result also holds for the 4-by-4, 16-QAM scenario used in other experiments. For the considered scenario, Theorem 1 states that the latent feature dimension corresponding to the compression lower bound is eight, while Theorem 2 gives a dimension of six matching the upper compression bound.

Figure 3 shows the end-to-end block error rate performance when the latent feature size is the only parameter that changes in a series of autoencoder models. In

^[1]<https://github.com/utcsilab/eq-net>

all cases, the soft bit matrix is estimated with the optimal ML algorithm, and used to trained and evaluate the quantization autoencoders, each with a different size of the latent feature representation. For reference, we also plot the performance of the ideal, uncompressed ML estimate. The following conclusions can be drawn from this plot:

- 1 The upper bound (in this case, corresponding to $\dim(\mathbf{z}) = 6$) is sufficient to achieve near-optimal performance, with minimal (less than 0.1 dB) performance losses. We posit that this departure from the ideal curve in Figure 3 is due to the residual training error. Several ideas on how to reduce this gap are discussed in [41], but we do not discuss them for simplicity.
- 2 Any attempt to further compress the soft bit matrix *beyond* this limit – that is, cases where $\dim(\mathbf{z}) < 6$ – is met with an increase of error and departure from ML performance, providing evidence that the compression ratio of Theorem 2 is optimal.

This result justifies the use of a latent feature space of dimension $3N_t$ in all the remaining experiments.

3.2 Importance of Two-Stage Training

To highlight the benefit of re-using the feature space learned during quantization for estimation, we investigate the difference in performance of the proposed two-stage learning approach against a single-stage learning baseline that combines (8) and (12). We use exactly the same architecture in both cases. For the single-stage baseline, we train the two networks jointly for a total of 500 epochs, with the learning rate divided by a factor of two if the validation loss does not improve for 100 consecutive epochs. For the proposed two-stage approach, each stage is trained for the same number of 500 epochs. We increase the starting learning rate for the joint approach to compensate for the untrained latent representation, based on the larger observed loss at initialization.

Figure 4 shows the validation loss for both methods in a 2-by-2, 64-QAM scenario under i.i.d. Rayleigh fading. This highlights the superior performance of the proposed two-stage method: baseline single-stage training is unstable and converges much slower, whereas the two-stage method is more stable, and converges to a better solution. While it is possible that the single-stage approach will eventually converge

(either through significantly longer training time, or by using a significantly larger neural network), this can be achieved much faster with the proposed approach, even when reducing the number of epochs of the two-stage approach by half.

3.3 Estimation Performance

We evaluate two variants of EQ-Net: EQ-Net-L (a low-latency architecture with one residual block) and EQ-Net-P (a high-performance architecture with three residual blocks). We compare our method with two state-of-the-art deep learning baselines: the scheme in [25], which we further refer to as NN-Det for brevity, and the OAMP-Net2 approach in [23]. We implement both deep learning baselines in Tensorflow and optimize their hyper-parameters. We additionally compare with two non-learning based approaches: ZF-SIC and SD. We use the default implementation of SD available in MATLAB, which does not support GPU vectorized execution, and we re-implement the ZF-SIC algorithm in Tensorflow. Details are available in our source code.

For 2-by-2 64-QAM, we train two variants of NN-Det: NN-Det-P (4 unfolded blocks) with the same end-to-end performance as our proposed method and NN-Det-L (3 unfolded blocks) which trades off some of the performance for lower end-to-end latency. For 4-by-4 16-QAM, we train a single NN-Det model with 10 unfolded blocks, labeled as high-performance. For OAMP-Net2, we only use the detector part with trainable step sizes and do not perform channel estimation, instead relying on exact CSI. At test time, for both NN-Det and OAMP-Net2 we estimate the soft bits using (7), since these methods output symbol probabilities.

We quantify end-to-end latency of a batch of B samples (including transfer to/from GPU where applicable) by measuring the average (using 10000 batches) execution times using the `timeit` Python module for all algorithms. For the CPU measurements, the processing is executed on a single thread, thus the latency figures show the potential for vectorized computation. The CPU is an Intel i9-9900x running at 3.5 GHz and the GPU is an NVIDIA RTX 2080Ti.

From Table 1 and Table 2, it is observed that, while SD is efficient for a batch size of 1, the degree of parallelism is much lower, explained by the heavy use of sorting and the tree search procedure [20]. ZF-SIC also enjoys a very low latency in both scenarios, but suffers from a large incurred performance drop. In contrast, all deep

Table 1 Latency benchmarks of evaluated soft-output MIMO detection algorithms for a 2-by-2 64-QAM scenario. B represents the batch size during parallelized inference. All values are expressed in milliseconds and represent average execution times for the entire batch.

	EQ-Net (L)	EQ-Net (P)	NN-Det (L)	NN-Det (P)	OAMP-Net2	ZF-SIC	SD
Latency CPU [ms], $B = 1$	1.511	1.903	2.269	2.689	2.125	0.749	0.6935
Latency CPU [ms], $B = 16$	1.767	2.292	2.565	3.073	2.376	1.050	4.547
Latency GPU [ms], $B = 8192$	6.523	8.767	12.61	15.91	25.94	5.311	—

Table 2 Latency benchmarks of evaluated soft-output MIMO detection algorithms for a 4-by-4 16-QAM scenario. B represents the batch size during parallelized inference. All values are expressed in milliseconds and represent average execution times for the entire batch.

	EQ-Net (L)	NN-Det (P)	OAMP-Net2	ZF-SIC	SD
Latency CPU [ms], $B = 1$	1.807	3.456	2.835	1.055	1.622
Latency CPU [ms], $B = 16$	2.256	3.953	3.227	1.200	19.52
Latency GPU [ms], $B = 8192$	9.107	17.251	30.501	6.253	—

learning-based approaches do not involve any sorting operations, leading to more efficient parallel implementations that favourably scale with the batch size.

Figure 5 and Figure 6 plots the estimation performance of the algorithms in a 2-by-2 64-QAM and 4-by-4 16-QAM scenario, respectively. For the 64-QAM scenario, the low latency (L) version of EQ-Net achieves the same performance as the high-performance (P) version of NN-Det. This is better highlighted in Figure 6, where we only compare EQ-Net (L) with NN-Det (P) and reach the same conclusion. The performance of EQ-Net also surpasses the ZF-SIC baseline, but there is still a gap with respect to the SD algorithm. However, the latency of EQ-Net is reduced compared to all deep learning baselines, as well as the SD baseline when evaluated on CPU for a batch size of at least $B = 16$.

The reduced latency compared to the other deep learning methods can be attributed to the fact that the baselines rely on unfolded detection approaches and require additional linear algebra computations which are sequential in nature, such as matrix inversion and conjugate-multiply after each iteration. In contrast, our approach does not involve any of these steps: EQ-Net uses the received symbols $\mathbf{y}$ and the estimated channel $\mathbf{H}$ and a deep neural network for direct feed-forward inference, without additional matrix operations. This fact becomes apparent when the algorithms are run on specialized inference hardware, such as GPUs. The last lines in both Table 1 and Table 2 highlight the increased latency gap when the algorithm is deployed in a scenario with a massive number of users, where a single base station performs computations in large batches. In this case, EQ-Net is competitive in terms of latency with the simple ZF-SIC baseline, which is also the most efficient in terms of latency due to its simple implementation and high degree of parallelization [42].

The number of parameters (weights) of EQ-Net is equal to 440k for EQ-Net-L, and three times more for EQ-Net-P, while NN-Det-L has a total of 195k parameters, while NN-Det-P uses 260k. In contrast, OAMP-Net2 does not use any deep neural networks and only has 16 trainable parameters (two for each of the eight unfolded blocks used). Despite the larger number of weights, EQ-Net-L is more efficient than NN-Det-P in terms of latency because of its limited depth, which dictates the critical path latency.

3.4 Quantization Performance

For the quantization task, we investigate the performance of EQ-Net against the maximum mutual information quantizer in [27] and the deep learning based scheme in [31], which is designed for SISO scenarios. We extend this baseline to MIMO scenarios by splitting the soft bit matrix into rows and quantizing each of them separately. This serves as a strong baseline that also reveals the quantization gains obtained by considering redundancy across the spatial dimension. For our approach, we train a separate k-means++ quantizer with 64 levels for each scalar dimension of the latent space, thus requiring six bits of storage for each latent component. For a 2-by-2 64-QAM scenario, this amounts to compressing the soft bit matrix of 12

values using a 36-bit codeword, leading to an effective storage cost of three bits per soft bit.

The results in Figure 7 show that EQ-Net is superior to both baseline methods and can efficiently compress the soft bit matrix using an equivalent storage cost of 3 bits per soft bit with a minimal performance loss. Compared to [31], EQ-Net achieves a 16% compression gain with the same end-to-end performance, whereas compared to [27] EQ-Net boosts the performance of the system by 0.65 dB while using same compression ratio. The difference in performance when compared to [31] highlights the importance of jointly learning a feature space across the spatial dimensions of MIMO channels.

EQ-Net also has the advantage of supporting a non-uniform quantization rate. For example, if five bits are used to store each dimension of the feature space, then the entire soft bit matrix can be stored using 30 values, with an effective cost of 2.5 bits per soft bit. In contrast, classical compression methods that are applied to each soft bit or row of soft bits individually would have to use an imbalanced compression scheme to achieve fractional rates, potentially leading to increased error floors in the system performance.

3.5 Robustness to Distributional Shifts

All the deep learning approaches so far have been tested on the same distributions as the one they were trained on. Practical scenarios may, however, involve either test-time distributional shifts (e.g., completely different fading models) or may be faced with imperfect channel state information during deployment. For soft bit estimation and quantization, the most fragile part of the system is given by the matrix $\mathbf{H}$ and any potential distributional mismatch that may occur. To evaluate the robustness of our approach, we consider models trained on channels from an i.i.d. Rayleigh fading model and tested – without *any* further adjustments or fine-tuning – on realizations of the CDL-A non line-of-sight channel model adopted by the 5G-NR specifications [43]. This is a realistic model that has a larger degree of spatial correlation than the Rayleigh model and is driven by physical propagation laws.

Figure 8 shows the quantization performance under this shift in a 2-by-2 64-QAM scenario, where EQ-Net-R stands for models trained on Rayleigh channels

and tested on CDL-A. We simulate CDL-A channels at a carrier frequency of 6 GHz, with a delay spread of 316 ns, corresponding to a street canyon scenario [44]. To accurately evaluate robustness, we also train a EQ-Net-5G model on $\mathbf{\Lambda}^{(\text{ML})}$ estimated from realizations of the CDL-A channel model. While a performance gap of around 1 dB is apparent between EQ-Net-5G and the ML solution, the EQ-Net-R scheme is robust and does not exhibit any error floors or performance loss, having the same performance as EQ-Net-5G. This is an indicator that the compressed feature representation learned using Rayleigh channels can be used in a wide variety of conditions, showing a stable performance when reconstructing the soft bits from different channel distributions and making it practical for relay scenarios, where different links may experience completely different channel distributions.

Figure 9 investigates the estimation performance under the same shift and reveals a higher degree of robustness compared to the baseline NN-Det approach, retaining a performance close to that of the ZF-SIC algorithm. While the two methods started from a very similar same end-to-end performance, as shown in Fig. 5, EQ-Net can avoid a severe error floor occurring early, even though this setup is challenging and does eventually lead to a saturated performance. We attribute this robustness to the bottleneck applied during training, as well as the particular expression of the weighted MSE loss used during training. Equation 11 increases the weight of uncertain soft bits, which are more likely to occur in scenarios with ill-conditioned channels, such as those from the CDL-A model. Overall, this result highlights the potential of robust feature learning for soft bit estimation and also poses learning soft bit estimation schemes in ill-conditioned difficult channels as an open problem.

Finally, we also investigate the performance of our approach in the case of CSI estimation impairments at the receiver. For this we use a corrupted version of $\mathbf{H}$ generated from the model $\hat{\mathbf{H}} = \mathbf{H} + \mathbf{N}$, where $\mathbf{N}$ is Gaussian noise with zero mean and covariance matrix $\sigma_{\text{CSI}} \mathbf{I}_{N_r \times N_t}$. This models impairments coming from the channel estimation module and, while not capturing hardware impairments, is a widely used model to test the robustness of downstream estimation algorithms [45, 46]. This corruption model serves as an easy robustness test of the network, since we can accurately control the amount of introduced distortion through the parameter σ_{CSI} . We choose a specific SNR value such that the two algorithms start from the same end-to-end performance and we vary σ_{CSI} . Figure 10 plots average per-

formance over 10000 channel realization with corrupted CSI and shows that the proposed algorithm is as robust as the NN-Det baseline to this type of impairment, while still benefiting from the much lower end-to-end latency in Table 1.

4 Conclusions

In this paper we have proposed a deep learning framework that jointly solves the tasks of soft bit estimation and quantization in MIMO scenarios. We have derived theoretical lower and upper bounds on the size of compressed representation of soft bits in MIMO channels (Theorems 1 and 2) and further showed evidence that the upper bound is practically achievable in arbitrary channels. Our approach has been shown to be practical in terms of latency and is compatible with any MIMO system, such as the MIMO-OFDM used in 5G scenarios, relaying scenarios, or distributed communication systems, which would benefit from both quantization and estimation gains. Throughout evaluation, our approach has shown superior performance, competitive distributional and impairment robustness to state-of-the-art deep learning based estimation methods.

One limitation of the proposed approach that remains is the presence of an error floor when faced with severe distributional shifts at test time, as per Figure 9. Even though our results show that this floor is much lower than that of the prior deep learning based work in [25], there is still room for improvement, at least in overcoming the ZF-SIC algorithm across the entire SNR range. For example, our method could easily be extended to account for perturbations during training or be trained on a dataset that pools together realizations of multiple channel models. Another promising direction for future work is removing the requirement of training a separate model for each MIMO configuration and leveraging flexible deep learning architecture to learn truly universal algorithms for soft bit processing.

List of Abbreviations

CDL	Clustered Delay Line
CPU	Central Processing Unit
CSI	Channel State Information
GPU	Graphical Processing Unit
HARQ	Hybrid Automatic Repeat Request
LDPC	Low-Density Parity Check

MIMO	Multiple Input Multiple Output
ML	Maximum Likelihood
MSE	Mean Squared Error
OAMP	Orthogonal Approximate Message Passing
OFDM	Orthogonal Frequency-Division Multiplexing
QAM	Quadrature Amplitude Modulation
SD	Sphere Decoding
SISO	Single Input Single Output
SNR	Signal-to-Noise Ratio
V-BLAST	Vertical-Bell Laboratories Layered Space-Time
ZF-SIC	Zero-Forcing with Successive Interference Cancellation

Declarations

Availability of data and materials

The software used to generate soft bit data for training and evaluation, as well as a complete implementation of the proposed approach and the baselines are publicly available in the EQ-Net Github repository, located at <https://github.com/utcsilab/eq-net>.

Competing interests

The authors declare that they have no competing interests.

Funding

MA, SV, and JT have been supported by grant ONR N00014-19-1-2590. The funding body has not played any role in the design of the study, analysis, interpretation of data, or writing the manuscript.

Authors' contributions

MA designed and implemented the training and evaluation software pipelines for the algorithms, and contributed in writing all sections. MA and JT formulated, proved and wrote the theoretical results section. All authors contributed in writing and revising the introduction and conclusion sections, and in defining the problem formulation. All authors read and approved the final manuscript.

Acknowledgements

The authors would like to thank Ian Roberts for providing early feedback for the manuscript.

References

1. Viswanathan, H., Mogensen, P.E.: Communications in the 6g era. *IEEE Access* **8**, 57063–57074 (2020)
2. Hagenauer, J.: Soft is better than hard. In: *Communications and Cryptography*, pp. 155–171. Springer, ??? (1994)
3. Larsson, E.G., Stoica, P., Li, J.: On maximum-likelihood detection and decoding for space-time coding systems. *IEEE Transactions on Signal Processing* **50**(4), 937–944 (2002)
4. Bennis, M., Debbah, M., Poor, H.V.: Ultrareliable and low-latency wireless communication: Tail, risk, and scale. *Proceedings of the IEEE* **106**(10), 1834–1853 (2018)
5. 3GPP: 38.214: Physical Layer Procedures for Data. (2021). 3GPP. Ver. 16.4
6. Albreem, M.A., Juntti, M., Shahabuddin, S.: Massive mimo detection techniques: A survey. *IEEE Communications Surveys & Tutorials* **21**(4), 3109–3132 (2019)
7. Zhao, B., Valenti, M.C.: Practical relay networks: a generalization of hybrid-arq. *IEEE Journal on selected areas in communications* **23**(1), 7–18 (2005)
8. Song, Y., Devroye, N.: Lattice codes for the gaussian relay channel: Decode-and-forward and compress-and-forward. *IEEE Transactions on Information Theory* **59**(8), 4927–4948 (2013)
9. Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A.N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al.: Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977* (2019)
10. Feng, S., Niyato, D., Wang, P., Kim, D.I., Liang, Y.-C.: Joint service pricing and cooperative relay communication for federated learning. In: *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 815–820 (2019). IEEE
11. Frenger, P., Parkvall, S., Dahlman, E.: Performance comparison of harq with chase combining and incremental redundancy for hsdpa. In: *IEEE 54th Vehicular Technology Conference. VTC Fall 2001. Proceedings (Cat. No. 01CH37211)*, vol. 3, pp. 1829–1833 (2001). IEEE
12. Zhang, C., Patras, P., Haddadi, H.: Deep learning in mobile and wireless networking: A survey. *IEEE Communications surveys & tutorials* **21**(3), 2224–2287 (2019)
13. Erpek, T., O'Shea, T.J., Sagduyu, Y.E., Shi, Y., Clancy, T.C.: Deep Learning for Wireless Communications (2020). [2005.06068](#)
14. Akhtar, N., Mian, A.: Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access* **6**, 14410–14430 (2018)
15. Kokalj-Filipovic, S., Miller, R.: Adversarial examples in rf deep learning: detection of the attack and its physical robustness. *arXiv preprint arXiv:1902.06044* (2019)
16. Hu, Q., Gao, F., Zhang, H., Jin, S., Li, G.Y.: Deep learning for channel estimation: Interpretation, performance, and comparison. *IEEE Transactions on Wireless Communications* **20**(4), 2398–2412 (2020)
17. Wolniansky, P.W., Foschini, G.J., Golden, G.D., Valenzuela, R.A.: V-BLAST: An architecture for realizing very high data rates over the rich-scattering wireless channel. In: *Proc. URSI Intl. Symp. on Signals, Syst., and Electron.*, pp. 295–300 (1998)
18. Shen, C., Zhu, Y., Zhou, S., Jiang, J.: On the performance of v-blast with zero-forcing successive interference cancellation receiver. In: *IEEE Global Telecommunications Conference, 2004. GLOBECOM'04.*, vol. 5, pp. 2818–2822. IEEE
19. Hassibi, B., Vikalo, H.: On the sphere-decoding algorithm i. expected complexity. *IEEE transactions on signal processing* **53**(8), 2806–2818 (2005)
20. Studer, C., Burg, A., Bolcskei, H.: Soft-output sphere decoding: Algorithms and VLSI implementation. *IEEE JSAC* **26**(2), 290–300 (2008)

21. Wu, Y., McAllister, J.: Configurable quasi-optimal sphere decoding for scalable mimo communications. *IEEE Transactions on Circuits and Systems I: Regular Papers* **68**(6), 2675–2687 (2021)
22. Mohammadkarimi, M., Mehrabi, M., Ardakani, M., Jing, Y.: Deep learning-based sphere decoding. *IEEE Transactions on Wireless Communications* **18**(9), 4368–4378 (2019)
23. He, H., Wen, C.-K., Jin, S., Li, G.Y.: Model-driven deep learning for mimo detection. *IEEE Transactions on Signal Processing* **68**, 1702–1715 (2020)
24. Ma, J., Ping, L.: Orthogonal AMP. *IEEE Access* **5**, 2020–2033 (2017)
25. Sholev, O., Permuter, H.H., Ben-Dror, E., Liang, W.: Neural network MIMO detection for coded wireless communication with impairments. In: *Proc. IEEE Wireless Commun. Netwk. Conf.*, pp. 1–8 (2020)
26. Shental, O., Hoydis, J.: Machine LLRning: Learning to softly demodulate. In: *Proc. IEEE Global Commun. Conf. Workshops*, pp. 1–7 (2019)
27. Winkelbauer, A., Matz, G.: On quantization of log-likelihood ratios for maximum mutual information. In: *Proc. IEEE Intl. Workshop on Signal Proc. Adv. in Wireless Commun.*, pp. 316–320 (2015)
28. Zeitler, G., Bauch, G., Widmer, J.: Quantize-and-forward schemes for the orthogonal multiple-access relay channel. *IEEE transactions on communications* **60**(4), 1148–1158 (2012)
29. Danieli, M., Forchhammer, S., Andersen, J.D., Christensen, L.P., Christensen, S.S.: Maximum mutual information vector quantization of log-likelihood ratios for memory efficient harq implementations. In: *2010 Data Compression Conference*, pp. 30–39 (2010). IEEE
30. Rave, W.: Quantization of log-likelihood ratios to maximize mutual information. *IEEE Signal Processing Letters* **16**(4), 283–286 (2009)
31. Arvinte, M., Tewfik, A.H., Vishwanath, S.: Deep log-likelihood ratio quantization. In: *2019 27th European Signal Processing Conference (EUSIPCO)*, pp. 1–5 (2019). IEEE
32. Tse, D., Viswanath, P.: *Fundamentals of Wireless Communication*. Cambridge University Press, ??? (2005)
33. Chen, J., Dholakia, A., Eleftheriou, E., Fossorier, M.P., Hu, X.-Y.: Reduced-complexity decoding of ldpc codes. *IEEE transactions on communications* **53**(8), 1288–1299 (2005)
34. Hanin, B.: Which neural net architectures give rise to exploding and vanishing gradients? In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 580–589 (2018)
35. Heath, R.W., Gonzalez-Prelcic, N., Rangan, S., Roh, W., Sayeed, A.M.: An overview of signal processing techniques for millimeter wave mimo systems. *IEEE journal of selected topics in signal processing* **10**(3), 436–453 (2016)
36. Sampath, H., Stoica, P., Paulraj, A.: Generalized linear precoder and decoder design for mimo channels using the weighted mmse criterion. *IEEE Transactions on Communications* **49**(12), 2198–2206 (2001)
37. Heo, B., Kim, J., Yun, S., Park, H., Kwak, N., Choi, J.Y.: A comprehensive overhaul of feature distillation. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1921–1930 (2019)
38. Arthur, D., Vassilvitskii, S.: k-means++ the advantages of careful seeding. In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1027–1035 (2007)
39. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014)
40. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: Tensorflow: A system for large-scale machine learning. In: *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283 (2016)
41. Arvinte, M., Vishwanath, S., Tewfik, A.H.: Deep learning-based quantization of l-values for gray-coded modulation. In: *2019 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6 (2019). IEEE
42. Husmann, C., Georgis, G., Nikitopoulos, K., Jamieson, K.: Flexcore: Massively parallel and flexible processing for large {MIMO} access points. In: *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17)*, pp. 197–211 (2017)
43. 3GPP: Study on Channel Model for Frequency Spectrum Above 6 GHz. (2019). 3GPP
44. 3GPP: 38.901: Study on Channel Model for Frequencies from 0.5 to 100 GHz. (2019). 3GPP. Ver. 16
45. Narasimhan, R.: Error propagation analysis of v-blast with channel-estimation errors. *IEEE Transactions on Communications* **53**(1), 27–31 (2005)
46. Miridakis, N.I., Tsiftsis, T.A.: On the joint impact of hardware impairments and imperfect csi on successive decoding. *IEEE Transactions on Vehicular Technology* **66**(6), 4810–4822 (2016)

Figure Title and Legend

Figure 1 High-level block diagram of the EQ-Net architecture. In the first training stage, and at test time for the soft bit quantization task, the dashed blocks are learned and used for inference, respectively. In the second training stage, and at test time for the soft bit estimation task, only the estimation encoder marked with solid lines is learned. Importantly, the decoder g_Q is *only* used during inference, and is kept frozen while training the estimation encoder f_E .

Figure 2 Internal architecture for the functional blocks in Figure 1 implemented as deep neural networks. (a) The quantization encoder. (b) The quantization decoder. (c) The estimation encoder. In all sub-figures, the blue blocks represent fully-connected layers, with the output dimension displayed. The yellow and green blocks represents the relu and $\tanh$ activations, respectively, while the purple block represents a concatenation operator.

Figure 3 Impact of latent feature dimension on different instances of EQ-Net without numerical quantization. Simulations are carried out in an i.i.d. Rayleigh fading, 2×2 64-QAM scenario with an (324, 648) LDPC code. A feature dimension of six corresponds to the upper bound in Theorem 2, which is proven explicitly for diagonalized channels, but is shown here to be achievable for arbitrary channels.

Figure 4 Validation loss for EQ-Net and a naïve single-stage supervised baseline. In both cases, the networks have the same architecture and initial weights. All weights receive exactly the same number of gradient updates in the two experiments. The plot includes shaded standard deviation areas over ten runs.

Figure 5 Estimation (MIMO detection) performance of EQ-Net versus state-of-the-art approaches. Simulations are carried out in a 2-by-2, 64-QAM Rayleigh fading and LDPC-coded scenario. Table 1 further compares the end-to-end latency of all methods.

Figure 6 End-to-end estimation performance of proposed algorithms against state-of-the-art approaches. Simulations are carried out in a 4-by-4, 16-QAM Rayleigh fading scenario. Table 2 further compares the end-to-end latency of all methods.

Figure 7 Quantization performance of EQ-Net evaluated against state-of-the-art methods. Simulations are carried out in a 2-by-2, 64-QAM, i.i.d. Rayleigh fading and LDPC-coded scenario.

Figure 8 Quantization performance of EQ-Net under a severe distributional shift. This shift is induced by training on an i.i.d. Rayleigh fading channel and testing on the 5G-NR CDL-A MIMO channel model for a 2-by-2 64-QAM LDPC-coded scenario.

Figure 9 Estimation performance of EQ-Net and the similarly performing NN-Det-P [25] under a severe distributional shift. This shift is induced by training both methods on i.i.d. Rayleigh fading channels and testing them on the 5G-NR CDL-A MIMO channel model for a 2-by-2 64-QAM LDPC-coded scenario, without any further fine-tuning.

Figure 10 Estimator robustness of EQ-Net and the similarly performing NN-Det-P [25] under imperfect CSI. Simulations are carried out in a 2-by-2 64-QAM LDPC-coded scenario. For better visualization, we only plot the evolution of block error rate at an SNR value of 14.35 dB as the CSI error increases. The two horizontal lines show the average end-to-end performance of the two algorithms with ideal CSI knowledge at the same SNR value.

Appendix

Proof of Theorem 1

Let the QR decomposition of $\mathbf{H}$ be given by $\mathbf{H} = \mathbf{Q}\mathbf{R}$, where $\mathbf{Q}$ satisfies $\mathbf{Q}^H\mathbf{Q} = \mathbf{I}_{N_t}$ and $\mathbf{R}$ is a square upper triangular matrix. This decomposition exists for any arbitrary channel, even if the matrix $\mathbf{H}$ is rank-deficient. Then, after left-side multiplication with $\mathbf{Q}^H$, the system in (1) becomes:

$$\tilde{\mathbf{y}} = \mathbf{Q}^H\mathbf{y} = \mathbf{R}\mathbf{x} + \mathbf{Q}^H\mathbf{n} = \mathbf{R}\mathbf{x} + \tilde{\mathbf{n}}.$$

Given that the rows of $\mathbf{Q}$ are orthonormal, it follows that $\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 = \|\tilde{\mathbf{y}} - \mathbf{R}\mathbf{x}\|_2^2$, for any $\mathbf{x}$. Then, by replacing all the norms in (7), it follows that:

$$\Lambda_{i,k}^{(\text{ML})} = \frac{\sum_{\mathbf{x} \in \mathcal{C}, b_{i,k}=1} \exp\left(-\frac{\|\tilde{\mathbf{y}} - \mathbf{R}\mathbf{x}\|_2^2}{\sigma_n^2}\right) + \sum_{\mathbf{x} \in \mathcal{C}, b_{i,k}=0} \exp\left(-\frac{\|\tilde{\mathbf{y}} - \mathbf{R}\mathbf{x}\|_2^2}{\sigma_n^2}\right)}{\sum_{\mathbf{x} \in \mathcal{C}, b_{i,k}=1} \exp\left(-\frac{\|\tilde{\mathbf{y}} - \mathbf{R}\mathbf{x}\|_2^2}{\sigma_n^2}\right) - \sum_{\mathbf{x} \in \mathcal{C}, b_{i,k}=0} \exp\left(-\frac{\|\tilde{\mathbf{y}} - \mathbf{R}\mathbf{x}\|_2^2}{\sigma_n^2}\right)}.$$

Thus, there exists a surjective functions that maps the features $\tilde{\mathbf{y}}/\sigma_n^2$ and $\mathbf{R}/\sigma_n^2$ to the matrix of soft bits $\mathbf{\Lambda}^{(\text{ML})}$ by implementing the equation above. In this case, the dimension of the domain of the function is given by the cost of storing these features. Since $\tilde{\mathbf{y}}$ is a complex-valued vector of length N_t it can be stored using $2N_t$ real-valued features. Further, $\mathbf{R}$ is a complex-valued upper triangular matrix of size $N_t \times N_t$ with real-valued elements on its diagonal meaning that it can be stored using N_t and $2\frac{(N_t-1)N_t}{2}$ real-valued features for its diagonal and off-diagonal

components, respectively. Thus, we immediately have that

$$2N_t + N_t + 2 \frac{(N_t - 1)N_t}{2} = N_t(N_t + 2)$$

real-valued features are sufficient to reconstruct $\mathbf{\Lambda}^{(\text{ML})}$ exactly. Thus, there exists a surjective function $g : \mathbb{R}^{N_t(N_t+2)} \rightarrow \mathbb{R}^{K \times N_t}$ such that $g(\tilde{\mathbf{y}}, \mathbf{R}) = \mathbf{\Lambda}^{(\text{ML})}$. The compression ratio of this function is equal to $R_{\text{low}} = \frac{KN_t}{N_t(N_t+2)} = \frac{K}{N_t+2}$, proving the lower bound.

Proof of Theorem 2

Let the singular value decomposition of $\mathbf{H}$ be given by $\mathbf{H} = \mathbf{U}\mathbf{S}\mathbf{V}^H$, where $\mathbf{U}$ and $\mathbf{V}$ both satisfy $\mathbf{U}^H\mathbf{U} = \mathbf{V}^H\mathbf{V} = \mathbf{I}_{N_t}$ and $\mathbf{S}$ is a real-valued, diagonal matrix of size $N_t \times N_t$.

Given that the transmitted symbols are $\mathbf{s} = \mathbf{V}\mathbf{x}$, the system in (1) takes the form:

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} = \mathbf{U}\mathbf{S}\mathbf{V}^H\mathbf{V}\mathbf{x} + \mathbf{n} = \mathbf{U}\mathbf{S}\mathbf{x} + \mathbf{n}.$$

Then, since $\mathbf{U}$ is orthonormal, left-multiplying with $\mathbf{U}^H$ leads to the post-processed received symbols:

$$\tilde{\mathbf{y}} = \mathbf{U}^H\mathbf{U}\mathbf{S}\mathbf{x} + \mathbf{U}^H\mathbf{n} = \mathbf{S}\mathbf{x} + \tilde{\mathbf{n}},$$

where $\tilde{\mathbf{n}}$ is still i.i.d. Gaussian noise since $\mathbf{U}$ is orthonormal. Finally, since $\mathbf{S}$ is diagonal, it follows that the soft bits corresponding to the i -th transmitted symbol only depend on the equation:

$$\tilde{y}_i = s_i x_i + \tilde{n}_i.$$

According to [31], the soft bits for each transmitted symbol in a scalar channel can be derived from a three-dimensional real-valued feature representation by explicitly storing the complex-valued $\tilde{y}$ and the complex-valued s_i/σ_n^2 . Thus, the soft bit matrix $\mathbf{\Lambda}^{(\text{ML})}$ can be represented using a $3N_t$ -dimensional feature representation when the channel is diagonalized. The compression ratio of this function is equal to $R_{\text{up}} = \frac{KN_t}{3N_t} = \frac{K}{3}$.

Given that $N_t \geq 1$, we have that R_{up} represents an upper compression bound for $K \geq 3$ and arbitrary channels $\mathbf{H}$, since at least the subset of soft bits derived from

diagonalized channels cannot be represented using fewer features without further assumptions.

Figures

Figure 1

High-level block diagram of the EQ-Net architecture. In the first training stage, and at test time for the soft bit quantization task, the dashed blocks are learned and used for inference, respectively. In the second training stage, and at test time for the soft bit estimation task, only the estimation encoder marked with solid lines is learned. Importantly, the decoder g_Q is only used during inference, and is kept frozen while training the estimation encoder f_E .

Figure 2

Internal architecture for the functional blocks in Figure 1 implemented as deep neural networks. (a) The quantization encoder. (b) The quantization decoder. (c) The estimation encoder. In all sub-figures, the blue blocks represent fully-connected layers, with the output dimension displayed. The yellow and green blocks represent the relu and tanh activations, respectively, while the purple block represents a concatenation operator.

Figure 3

Impact of latent feature dimension on different instances of EQ-Net without numerical quantization. Simulations are carried out in an i.i.d. Rayleigh fading, 2×2 64-QAM scenario with an (324, 648) LDPC

code. A feature dimension of six corresponds to the upper bound in Theorem 2, which is proven explicitly for diagonalized channels, but is shown here to be achievable for arbitrary channels.

Figure 4

Validation loss for EQ-Net and a naïve single-stage supervised baseline. In both cases, the networks have the same architecture and initial weights. All weights receive exactly the same number of gradient updates in the two experiments. The plot includes shaded standard deviation areas over ten runs.

Figure 5

Estimation (MIMO detection) performance of EQ-Net versus state-of-the-art approaches. Simulations are carried out in a 2-by-2, 64-QAM Rayleigh fading and LDPC-coded scenario. Table 1 further compares the end-to-end latency of all methods.

Figure 6

End-to-end estimation performance of proposed algorithms against state-of-the-art approaches. Simulations are carried out in a 4-by-4, 16-QAM Rayleigh fading scenario. Table 2 further compares the end-to-end latency of all methods.

Figure 7

Quantization performance of EQ-Net evaluated against state-of-the-art methods. Simulations are carried out in a 2-by-2, 64-QAM, i.i.d. Rayleigh fading and LDPC-coded scenario.

Figure 8

Quantization performance of EQ-Net under a severe distributional shift. This shift is induced by training on an i.i.d. Rayleigh fading channel and testing on the 5G-NR CDL-A MIMO channel model for a 2-by-2 64-QAM LDPC-coded scenario.

Figure 9

Estimation performance of EQ-Net and the similarly performing NN-Det-P [25] under a severe distributional shift. This shift is induced by training both methods on i.i.d. Rayleigh fading channels and testing them on the 5G-NR CDL-A MIMO channel model for a 2-by-2 64-QAM LDPC-coded scenario, without any further fine-tuning.

Figure 10

Estimator robustness of EQ-Net and the similarly performing NN-Det-P [25] under imperfect CSI. Simulations are carried out in a 2-by-2 64-QAM LDPC-coded scenario. For better visualization, we only plot the evolution of block error rate at an SNR value of 14.35 dB as the CSI error increases. The two horizontal lines show the average end-to-end performance of the two algorithms with ideal CSI knowledge at the same SNR value.