Arvinte *et al. J Wireless Com Network* (2022) 2022:51
https://doi.org/10.1186/s13638-022-02129-z

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

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

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 (E) and quantization (Q) in high-dimensional multiple-input multiple-output (MIMO) systems. We propose a two-stage algorithm that uses soft bit quantization as pretraining for estimation and is motivated by a theoretical analysis of soft bit representation sizes 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 also demonstrate that the quantization approach is feasible in single-user MIMO scenarios of up to 64×64 and can be used with different soft bit estimation algorithms than the ones during training. We investigate the robustness of the proposed approach and demonstrate that the model is robust to distributional shifts when used for soft bit quantization and is competitive with state-of-the-art deep learning approaches when faced with channel estimation errors in 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 soft bit quantization. Using soft—instead of hard—bit estimates for decoding error-correcting codes is known to provide an order of magnitude order in a reduction in end-to-end error rates [2], and 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. Starting with soft bit quantization using deep learning, we develop a framework that can address the two tasks simultaneously by

reusing the information learned during quantization for better supervised training of a soft bit estimation algorithm.

Soft bit quantization is required when storing a large number of soft bits for a potentially long duration of time, such as in hybrid automatic repeat request (HARQ) schemes [3] or when the soft bits themselves need to be forwarded across wireless environments, such as in relay [4] or fronthaul [5] systems. For example, in distributed communications systems [6], low-resolution soft bit quantization is required whenever relaying or feedback is involved, because system capacity represents a bottleneck [7], and soft bits must be transmitted without errors, using a low communication overhead. 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 [8, 9] to boost performance, making storage a potential system bottleneck.

Soft bit estimation in MIMO systems is a challenging practical problem due to the exponential complexity of the optimal solution [10] and stringent latency requirements in 5G-and-beyond communication systems [11]. For example, given a single-user MIMO (SU-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 [12], each of these requiring an expensive algorithm. Near-optimal estimation algorithms are a central part of end-to-end performance in coded systems (e.g., that use low-density parity-check (LDPC) codes) [13], 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 moderately sized SU-MIMO systems.

Deep learning methods have emerged as promising candidates for aiding or completely replacing signal processing blocks in MIMO communication systems [14, 15]. 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, as they are modular and compatible with existing communication protocols. While previous solutions have been developed for both tasks, there is currently no solution that addresses soft bit quantization in large MIMO systems and that tackles soft bit estimation and quantization in the same framework. Furthermore, the robustness of deep learning-based methods is still an open problem in the broader machine learning field [16, 17] and has been recognized as an issue in digital communications as well [18, 19]. 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 classical approaches under distributional shifts.

1.1 Related work

1.1.1 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 [20] 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 [21] 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 [22] extends the data-driven approach in [23] and proposes a vector extension to a maximum mutual information quantizer, but loses theoretical optimality guarantees. The method in [24] 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. Finally, prior work does not discuss or exploit the learned representations from the quantization task when maximum likelihood (ML) soft bits are used for training, which is also a component of EQ-Net.

1.1.2 Soft bit estimation

There are a broad number of classical (non-learning-based) signal processing algorithms that have been developed for soft bit estimation, 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 [13]. The V-BLAST algorithm [25] uses the idea of sequential estimation, with subsequent work leading to zero-forcing with successive interference cancelation (ZF-SIC) [26] as an algorithm for efficient detection, and the minimum mean square error with successive interference cancelation (MMSE-SIC) [27] to further improve performance. In these methods, the system is reduced to a (regularized) 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 low-complexity, but also low-performance methods, that may exhibit error floors. Sphere decoding [28, 29] formulates the detection problem as a tree search algorithm and performs a greedy search. In the soft-output version [29], multiple candidate solutions are used to estimate log-likelihoods. A drawback of this class of algorithms is the need for specialized hardware to accommodate latency budgets in 5G-and-beyond scenarios [30]. To address this, recent work has combined sphere decoding with deep learning for search radius and branch prediction [31].

More recently, there has been work in model-based approaches for MIMO detection, where differentiable optimization steps are interleaved with deep learning models. The work in [32] proposes OAMP-Net2 as a data-based extension to the orthogonal approximate message passing (OAMP) detection algorithm [33], 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 [34] 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 [35] 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 Sect. 3.2.

1.2 Contributions

Our contributions in this work are the following:

1 We prove lower and upper representation size bounds (Theorems 1 and 2) for the ML and MMSE-SIC soft bit estimates in arbitrary MIMO channels. The upper bound for ML is proved constructively through a construction that holds for any channel, while the lower bound for ML comes from the class of diagonalized MIMO channels. In practice, we verify that the lower bound for ML can be achieved *for arbitrary channels* and learned using deep neural networks, which becomes a design criterion in the proposed approach. For the MMSE-SIC algorithm, we prove that the lower bound is achievable in arbitrary channels.
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, reusing the quantization decoder. We experimentally verify that the two-stage training algorithm improves convergence and has better end-to-end coded system performance than the single-stage supervised training baseline when using small depth (shallow) networks.
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), different train–test distributions of the channel conditions and distributions of the input in large-scale MIMO scenarios (up to 64×64 SU-MIMO), when evaluated on the quantization task, and is competitive with other deep learning methods on the robust estimation task in smaller-sized MIMO scenarios of up 4×4.

2 Methods

2.1 System model

We assume a narrowband, instantaneous, digital, single-cell uplink 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 [36, Eq. 7.55]:

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}, \tag{1}$$

where $\mathbf{x} \in \mathbb{C}^{N_\mathrm{t}}$ is a vector of transmitted symbols drawn from a finite, discrete constellation $\mathcal{C}$, and $\mathbf{y} \in \mathbb{C}^{N_\mathrm{r}}$ is a vector of received symbols. N_t and N_r represent the number of transmitted and received data streams, respectively, and $\mathbf{n} \in \mathbb{C}^{N_\mathrm{r}}$ is an i.i.d. complex Gaussian noise vector with covariance matrix $\sigma_n^2\mathbf{I}$. For the remainder of this work, we deal with spatial multiplexing scenarios, where $N_\mathrm{r} \geq N_\mathrm{t}$ 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 kth log-likelihood ratio of the ith transmitted symbol is defined as [20]:

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

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 [20] and are used in practical decoders for error-correcting codes due to operations being simpler to carry out in the hyperbolic tangent domain [37]:

$$\Lambda_{i,k} = \tanh \frac{L_{i,k}}{2}. \tag{3}$$

In scenarios where the noise is nonzero, soft bits are constrained to the interval $(-1, 1)$, with large magnitude values tending to near-certain soft bits. This is beneficial for data-driven methods, because a limited dynamic range is also helpful in the stability of deep learning methods, where unbounded inputs can lead to exploding gradients [38]. By replacing the definition of the log-likelihood ratio from (2), we obtain that the soft bits are defined as

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

We organize the soft bits corresponding to a single MIMO channel use in the soft bit matrix $\Lambda \in \mathbb{R}^{N_\mathrm{t} \times K}$.

2.1.1 ML soft bits

The likelihoods $P(\mathbf{y}|b_{i,k})$ in (4) can be expanded by the total probability law. For the soft bit corresponding to the ith transmitted stream and the kth bit position, let $\bar{b}_{i,k} = \{b_{j,l}|(j,l) \neq (i,k)\}$ be the set of all remaining bit positions. The conditional probability of the (i, k)th bit position is given by

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

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. Because 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):

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

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 term in the sum in (6) corresponds to the conditional probability of the observed symbols given a specific transmitted vector 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

$$\Lambda_{i,k}^{(\mathrm{ML})} = \frac{\sum\limits_{\mathbf{x} \in \mathcal{C} | b_{i,k} = 1} \exp\left(-\frac{\left\|\mathbf{y} - \mathbf{H}\mathbf{x}\right\|_2^2}{\sigma_n^2}\right) - \sum\limits_{\mathbf{x} \in \mathcal{C} | b_{i,k} = 0} \exp\left(-\frac{\left\|\mathbf{y} - \mathbf{H}\mathbf{x}\right\|_2^2}{\sigma_n^2}\right)}{\sum\limits_{\mathbf{x} \in \mathcal{C} | b_{i,k} = 1} \exp\left(-\frac{\left\|\mathbf{y} - \mathbf{H}\mathbf{x}\right\|_2^2}{\sigma_n^2}\right) + \sum\limits_{\mathbf{x} \in \mathcal{C} | b_{i,k} = 0} \exp\left(-\frac{\left\|\mathbf{y} - \mathbf{H}\mathbf{x}\right\|_2^2}{\sigma_n^2}\right)}. \tag{7}$$

2.1.2 MMSE-SIC soft bits

The exact evaluation of (7) is computationally prohibitive even in moderately sized systems. We consider the MMSE-SIC algorithm based on the QR decomposition in [39], with a random order, which sequentially estimates the data streams and their corresponding soft bits. The channel matrix $\mathbf{H}$ is first augmented by adding the extra rows:

$$\underline{\mathbf{H}} = \left[\mathbf{H}; \sigma_n^2 \mathbf{I}_{N_t}\right], \tag{8}$$

where $\mathbf{I}_{N_t}$ is a square identity matrix of size $N_t \times N_t$. The QR decomposition of the augmented matrix is carried out as $\underline{\mathbf{H}} = \underline{\mathbf{Q}}\underline{\mathbf{R}}$, where $\underline{\mathbf{Q}}$ is an unitary matrix and $\underline{\mathbf{R}}$ is upper triangular. The first N_r rows of $\underline{\mathbf{Q}}$ are denoted as

$$\mathbf{Q} = \underline{\mathbf{Q}}_{1:N_r,:}. \tag{9}$$

The matrix $\mathbf{Q}$ is used to equalize the received symbols $\mathbf{y}$ as

$$\tilde{\mathbf{y}} = \mathbf{Q}^{\mathrm{H}}\mathbf{y} = \underline{\mathbf{R}}\mathbf{x} + \tilde{\mathbf{n}}, \tag{10}$$

where we used that $\underline{\mathbf{Q}}$ is unitary, and $\tilde{\mathbf{n}}$ represents the filtered Gaussian noise, with zero mean and standard deviation of the ith entry given by the norm of the ith column of $\mathbf{Q}$.

As the equivalent MIMO system in (10) is now upper triangular, sequential detection of the symbols and soft bits starts from the N_{t}th data stream, with the scalar system equation:

$$\tilde{y}_{N_{\mathrm{t}}} = \underline{r}_{N_{\mathrm{t}},N_{\mathrm{t}}} x_{N_{\mathrm{t}}} + \tilde{n}_{N_{\mathrm{t}}}, \tag{11}$$

and the corresponding soft bits are obtained by marginalization only over the bits of the ith symbol, via (5). This populates the N_{t}th row of the matrix $\Lambda^{(\mathrm{MMSE-SIC})}$, and detection continues sequentially, by subtracting all the previously estimated hard symbols from the estimated vector $\mathbf{x}^{(\mathrm{MMSE-SIC})}$. The generic scalar system equation for the ith stream is given by

$$\hat{y}_i = \tilde{y}_i - \sum_{j=i+1}^{N_{\mathrm{t}}} \underline{r}_{i,j} x_j^{(\mathrm{MMSE-SIC})} = \underline{r}_{i,i} x_i + \hat{n}_i, \tag{12}$$

where $\hat{n}_i$, in general, includes the remaining interference terms caused by incorrect detection of the previous streams. Assuming correct detection, we have that $\hat{n}_i = \tilde{n}_i$, and (5) can be used to estimate the soft bits of the current stream. Finally, the ZF-SIC algorithm has a similar flow, with the difference that $\underline{\mathbf{H}} = \mathbf{H}$, i.e., no matrix augmentation is performed before the QR decomposition.

2.1.3 LMMSE soft bits

In extremely large MIMO systems, the MMSE-SIC estimate may not be available even for quantization, because of the sequential nature of the algorithm in (12). The linear minimum mean squared error (LMMSE) algorithm uses the matrix $\mathbf{G} = \left(\mathbf{H}^{\mathrm{H}}\mathbf{H} + \sigma_n^2 \mathbf{I}_{N_{\mathrm{t}}}\right)^{-1}\mathbf{H}^{\mathrm{H}}$ to equalize the received symbol vector as

$$\tilde{\mathbf{y}} = \mathbf{G}\mathbf{y}. \tag{13}$$

Assuming that the off-diagonal terms of $\mathbf{GH}$ are negligible, the MIMO system can be decomposed in N_{t} parallel SISO systems, where the ith system equation is

$$\tilde{y}_i = g_i x_i + \tilde{n}_i, \tag{14}$$

where g_i is the ith diagonal element of $\mathbf{GH}$, and $\tilde{n}_i$ is Gaussian noise with zero mean and standard deviation given by the norm of the ith row of $\mathbf{G}$. Estimating the soft bit matrix $\Lambda^{(\mathrm{LMMSE})}$ is then done separately for each row, using the marginalization only over the bits of the ith symbol, via (5).

2.2 Soft bit quantization

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

- An encoder f_{Q} that compresses the matrix $\Lambda^{(\mathrm{ML})}$ 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 [40]).
- A decoder g_{Q} that recovers $\Lambda^{(\mathrm{ML})}$ as accurately as possible from a given bit stream, with respect to a pre-defined error function.

2.3 Soft bit estimation

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_n and outputs an estimate of the soft bit matrix $\tilde{\Lambda}$. While (7) is the optimal estimator for the soft bits when the noise is Gaussian and independent—a common situation in practice [40]—the two distinct sums in (7) each involve a number of $2^{N_{\mathrm{t}}K-1}$ terms. This leads to a prohibitive computational complexity even for moderate values of N_{t} and K, and algorithms that approximate the solution $\Lambda^{(\mathrm{ML})}$ by reducing the amount of performed computations are the main goal in efficient soft bit estimation.

2.4 Distortion metric for soft bits

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 desired soft bits 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 [24] to MIMO scenarios:

$$\mathcal{L}\left(\tilde{\Lambda}, \Lambda^{(\mathrm{ML})}\right) = \sum_{i=1}^{N_{\mathrm{t}}} \sum_{k=1}^{K} \frac{\left(\tilde{\Lambda}_{i,k} - \Lambda_{i,k}^{(\mathrm{ML})}\right)^2}{\left|\Lambda_{i,k}^{(\mathrm{ML})}\right| + \epsilon}, \tag{15}$$

where $\tilde{\Lambda}$ is the output of the estimator. $\Lambda^{(\mathrm{ML})}$ 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 [37], we use a value of $\epsilon = 0.001$ to bound the loss in (15), chosen as a trade-off between stability and accuracy.

2.5 Theoretical results

In this section, we prove lower and upper bounds on the representation size ratio of the soft bit matrices of the ML and MMSE-SIC algorithms. We define the representation size ratio of a surjective mapping $f : \mathbb{R}^n \to \mathbb{R}^m$ 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 achieving 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 Sects. 3.4 and 3.6, numerical quantization is applied to the feature representation, and its impact is evaluated.

The two bounds derived in this section are helpful in choosing the feature dimension in a deep autoencoder 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. For the ML soft bits, we prove that the lower bound is achievable for arbitrary channels by constructing a feature representation that achieves it. We prove the upper bound through a special class of diagonalized channels, and in practice we attempt, and are successful, in learning a deep autoencoder that achieves the upper bound for arbitrary channels in Sect. 3.1. For the MMSE-SIC soft bits, we prove that the upper bound can be achieved constructively, for arbitrary channels.

2.5.1 Lower representation size ratio bound for the ML estimate in arbitrary channels

We directly work with (1) and make no assumptions on $\mathbf{x}$, other than it being sampled from the constellation $\mathcal{C}$. We prove the following lower bound for the representation size 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* $f(\mathbf{z}) = \Lambda^{(\mathrm{ML})}$ *and a representation size ratio of* $R_{\mathrm{low,ML}} = K/(N_t + 2)$ *is achieved.*

The proof is provided in the Appendix and relies on the QR decomposition of $\mathbf{H}$. 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 matrix and the received symbols is sufficient for exact soft bit recovery using (4).

2.5.2 Upper representation size ratio bound for 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 [41] 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* $f(\mathbf{z}) = \Lambda^{(\mathrm{ML})}$ *and a representation size ratio of* $R_{\mathrm{up,ML}} = K/3$ *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 $\mathbf{U}$. As 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 [24].

2.5.3 Upper representation size ratio bound and achievability for the MMSE-SIC estimate in arbitrary channels

Theorem 3 *Let* $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$ *be the noisy received symbols in arbitrary channels. Then, there exists a surjective function* $f : \mathbb{R}^{3N_t} \rightarrow \mathbb{R}^{K \times N_t}$ *such that* $f(\mathbf{z}) = \Lambda^{(\text{MMSE-SIC})}$ *and a representation size ratio of* $R_{\text{up,MMSE}-\text{SIC}} = K/3$ *is achieved.*

The proof for this upper bound is provided in the Appendix. Intuitively, once the system is represented in its upper triangular form, we inductively prove that representing each row of the soft bit matrix can be done using a three-dimensional vector, regardless of K.

2.5.4 Design implications of the theoretical results

In practice, the lower bound of $K/(N_t + 2)$ is too weak to use for compressing the ML soft bits, as it is possible that $N_t \geq K - 2$. The upper bound, however, 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). 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 [24]. However, achieving the upper bound is non-trivial in arbitrary channels in the case of ML soft bits. A key component of EQ-Net is to assume that the upper bound for the ML soft bits can be achieved in arbitrary—and not only in diagonalized—MIMO channels. This is verified and discussed in Sect. 3.1. While we do not derive such a representation explicitly for the ML case, 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 suitable choice, where we set the dimension of the bottleneck feature representation to $3N_t$, regardless of the algorithm used to estimate the soft bits.

2.6 EQ-Net: joint estimation and quantization

EQ-Net is a supervised method that uses a deep autoencoder and leverages compression as a supervised pretraining task for learning a soft bit estimator: When training the estimator, we do not learn a direct mapping between received symbols and 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 for both ML and MMSE-SIC in Theorems 2 and 3, respectively).
2. In the second stage, we train an additional estimation encoder that takes the received symbols and CSI as input, and is trained to predict the features of the soft bits, obtained from the first stage. 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 Fig. 1. The ablation experiments in Sect. 3.2 show that the two-stage procedure—where we first train the pair f_Q and g_Q, and then reuse f_Q to train f_E—is an essential step when training a model with limited

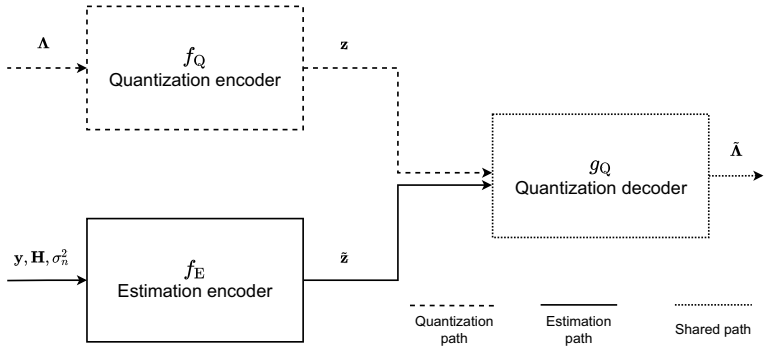

Fig. 1 High-level block diagram of the EQ-Net architecture. In the first training stage, and at test time for the soft bit quantization task, the dashed blocks are learned and used for inference, respectively. In the second training stage, the decoder g_{Q} is kept frozen while training the estimation encoder f_{E}. During deployment, the top or bottom signal paths are used to output the reconstructed (for quantization) or estimated (for estimation) soft bits $\tilde{\Lambda}$, respectively

depth and width, and that single-stage end-to-end training falls into unwanted local minima.

2.6.1 Stage 1: Compression and quantization

In the first stage, we train an autoencoder consisting of a quantization encoder f_{Q} and decoder g_{Q}. The input to the quantization encoder is the ML estimate of the soft bits $\Lambda^{(\mathrm{ML})}$ or, when infeasible due to the large complexity, the MMSE-SIC or LMMSE estimates of the soft bits. The parameters of the quantization encoder and decoder are denoted by θ_f and θ_g, respectively. The features output by the encoder are

$$\mathbf{z} = f_{\mathrm{Q}}\left(\Lambda; \theta_f\right), \tag{16}$$

where Λ denotes either $\Lambda^{(\mathrm{ML})}$, $\Lambda^{(\mathrm{MMSE\text{-}SIC})}$, or $\Lambda^{(\mathrm{LMMSE})}$. Let $\tilde{\Lambda}$ denote the matrix of reconstructed soft bits as

$$\tilde{\Lambda} = g_{\mathrm{Q}}\left(\mathcal{Q}(\mathbf{z}); \theta_g\right), \tag{17}$$

where $\mathcal{Q}$ is a quantization 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 word.

During training, we replace the $\mathcal{Q}$ operator with a differentiable approximation to obtain useful gradients by following previous work [24] 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 and to control the effective numerical precision of the features during training. The choice of approximating quantization noise with i.i.d. Gaussian noise is motivated both by the ease of implementation (the original motivation in [24]), and by the recent work in [42], that connects the variance of the added noise to the quantization error. In particular, choosing $\sigma_u^2 = 0.001$ leads to a feature resolution of approximately ten bits. While this is a finer quantization resolution than used in Sect. 3.4 (where we use as few as four bits per feature), it allows for flexibility in using a larger number of bits per feature, if desired, whereas injecting more noise during the training stage would remove this possibility.

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

$$\mathcal{L}_{\mathrm{Q}}(\theta_f, \theta_g) = \frac{1}{B}\sum_{n=1}^{B} \mathcal{L}(\tilde{\Lambda}^{(n)}, \Lambda^{(n)}), \tag{18}$$

where B is the batch size and $\mathcal{L}$ is given by (15), applied element-wise and summed. Once the quantization autoencoder is trained, the feature representation $\mathbf{z}$ is obtained for all training samples by performing a forward pass with the encoder f_{Q}.

2.6.2 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 the coherent 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

$$\mathcal{L}_{\mathrm{E}}(\theta_e) = \frac{1}{B}\sum_{n=1}^{B} \left\| f_{\mathrm{E}}\left(\mathbf{y}^{(n)}, \mathbf{H}^{(n)}, \sigma_n^{(n)}; \theta_e\right) - \mathbf{z}^{(n)} \right\|_1. \tag{19}$$

The choice of using an ℓ_1-loss for the feature loss in the estimation phase is empirical and investigated in Sect. 3.3, where it is compared to the more conventional mean squared error (MSE) loss.

Note that the ground truth soft bit matrix $\Lambda^{(\mathrm{ML})}$ is never explicitly used as a target output in the second stage, and we instead rely on the pretrained feature extractor given by f_{Q}. Obtaining the estimated soft bits from the estimated feature representation is straightforward during inference and is done as

$$\tilde{\Lambda} = g_{\mathrm{Q}}\left(f_{\mathrm{E}}(\mathbf{y}, \mathbf{H}, \sigma_n)\right). \tag{20}$$

A simple single-stage baseline is given by training f_{E} and g_{Q} together (as a single deep neural network with a bottleneck size of $3N_{\mathrm{t}}$), without first constructing the features $\mathbf{z}$. This is done by using the loss in (18) between the reconstructed soft bits in (20) and the ground truth soft bits Λ. The two-stage approach has the following advantages over this baseline:

- 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 Sect. 3.2.
- 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, as shown by the results in Sect. 3.3.

2.6.3 Implementation details

The models f_{Q}, f_{E} and g_{Q} are deep neural networks that use fully connected layers. A detailed block diagram of the models is shown in Fig. 2. All models use the rectified

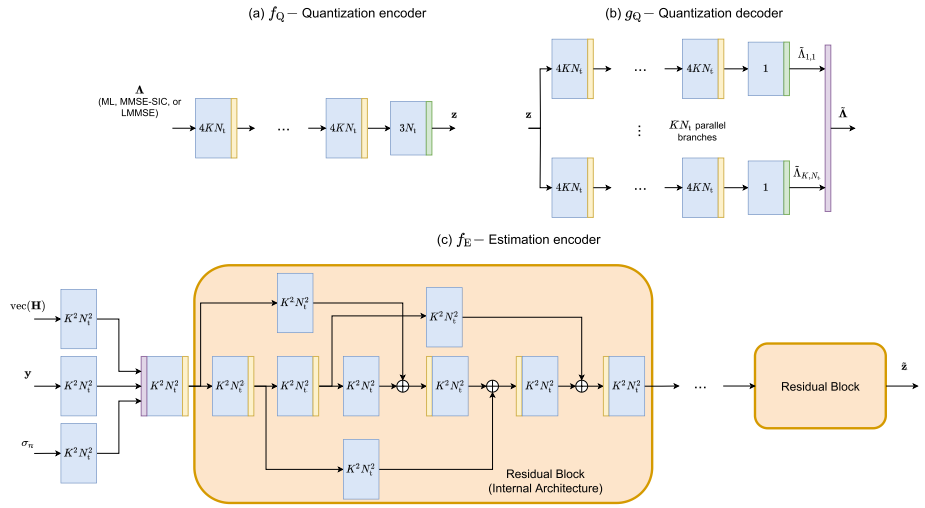

Fig. 2 Internal architecture for the deep neural networks in Figure 1. The blue blocks are fully connected layers, with the output dimension displayed. The yellow and green blocks represent the relu and tanh activations, respectively, while the purple block represents a concatenation operator. **a** The architecture of the quantization encoder f_{Q}. **b** The architecture of the quantization decoder g_{Q}. **c** The architecture of the estimation encoder f_{E}, composed of (potentially) multiple residual blocks. The internal architecture of a single residual block is shown here

linear unit activation function in the hidden layers, given by $\mathrm{relu}(x) = \max\{x, 0\}$, with the exceptions shown in Fig. 2.

The quantization encoder f_{Q} uses a simple feed-forward, fully connected deep neural network with six hidden layers, a width of $4N_{\mathrm{t}}K$ for the hidden layers, and $3N_{\mathrm{t}}$ 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 taken to increase the representational power of the network without sacrificing latency in higher-order modulation scenarios. The quantization decoder g_{Q} uses the branched architecture in Fig. 2b: The latent representation is separately processed using KN_{t} parallel, feed-forward, fully connected deep neural networks, each with six hidden layers and the same width as the encoder.

To numerically quantize the latent feature representation during testing, a factorized quantization codebook for the features $\mathbf{z}$ is learned by separately applying a scalar quantization function to each of its entries. We use the same data-based approach as [24] and train a k-means++ [43] scalar quantizer after f_{Q} and g_{Q} have been trained. We learn a separate quantizer for each latent feature. 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 $(\mathbf{y}, \mathbf{H}, \sigma_n)$ and uses their flattened, real-valued (obtained by concatenating the real and imaginary parts) versions with a 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 dimensions and scales. A single residual block of the estimation encoder contains an additional six hidden layers with residual connections between them, and is expanded in Fig. 2c.

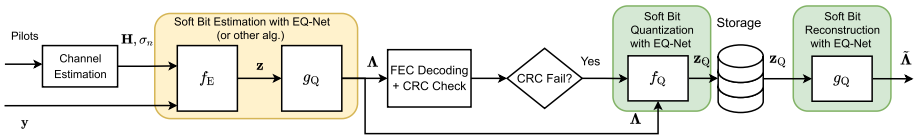

Fig. 3 Block diagram showing the deployment of EQ-Net in a MIMO receiver system architecture. The model g_{Q} is shared between the estimation and quantization functionalities

We use the Adam optimizer [44] with a batch size of 32768 samples, learning rate of 0.001 and default TensorFlow [45] parameters for both stages. Training, validation and test data consist of pre-generated ML or MMSE-SIC (in high-dimensional MIMO) soft bit estimates using (4) from 10000, LDPC-coded packets of size (324, 648), at six logarithmically spaced SNR values, dependent on the system size. The bits in a codeword are modulated, grouped in MIMO vectors, and transmission is simulated across different (randomly chosen) MIMO channels during training. The receiver uses either ML or MMSE-SIC algorithm to estimate the soft bits, which are used to train f_{Q} and g_{Q}. The data are split using a 80/10/10 train/validation/test ratio. We investigate the following SU-MIMO scenarios:

1 2×2, 64-QAM and 4×4, 16-QAM, both trained using ML soft bits. Our findings in Sects. 3.3 and 3.4 show that EQ-Net is viable for both estimation and quantization, respectively.
2 8×8, 16-QAM and 16×16, 16-QAM, both trained using the MMSE-SIC soft bits. We find that EQ-Net achieves state-of-the-art performance for soft bit quantization in Sect. 3.5, and discuss the limitations of using estimation in Sect. 3.8.
3 32×32, 16-QAM and 64×64, 16-QAM, where we investigate the performance of applying the models trained on 16×16, MMSE-SIC scenarios to larger sizes, and the LMMSE estimate, in quantization mode. Results are presented in Sect. 3.5.

Publicly available implementations can be found in the online code repository[1].

2.7 Operating modes

Training the components of EQ-Net is done offline, by first collecting (or simulating, if accurate environment models are available) a dataset of training ML soft bits Λ, the corresponding samples, and CSI information. Estimating the CSI information itself will require pilot symbols [46], but EQ-Net training is agnostic to their structure or number. Once this dataset is collected, the two phases described in Sect. 2.6 are applied in sequence to train the models in EQ-Net.

Figure 3 shows the role of EQ-Net during receiver deployment with a HARQ protocol, and how the three blocks in Fig. 1 are used. In *estimation* mode, the algorithm uses $\mathbf{y}, \mathbf{H}$ and σ_n as inputs for the estimation encoder f_{E}, followed by the decoder g_{Q}, and outputs the estimated soft bit matrix Λ. As the results in Fig. 4 have showed, inaccurate CSI may, in general, degrade the performance of the EQ-Net soft bit estimation module. While not explored in this work, further training (adapting) the estimation module

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

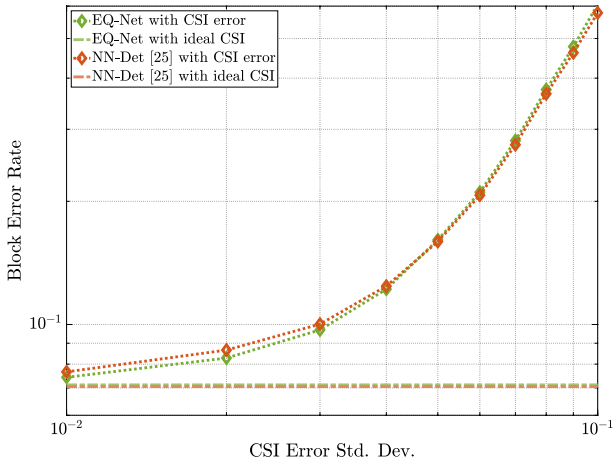

Fig. 4 Estimator robustness of EQ-Net (L), and the similarly performing NN-Det (P) under imperfect CSI for 2 × 2, 64-QAM, i.i.d. Rayleigh fading, and LDPC-coded with size (324, 648). The plot shows the coded block error rate at SNR=17.3 dB as a function of the CSI error. The two horizontal lines show the average end-to-end performance of the two algorithms with ideal CSI knowledge at the same SNR value

during deployment to account for inaccurate CSI using a robust optimization objective [47] is possible.

Any external soft bit estimation algorithm (e.g., ML, or MMSE-SIC) can be used, instead of the EQ-Net estimation encoder. This does not preclude EQ-Net being used for *quantization*, and the modes do not need to be used together, even though the model g_{Q} is shared between them. If codeword decoding fails, the soft bits from the failed transmission must be stored, as shown in Fig. 3. If the soft bit estimation module outputs log-likelihood ratios instead of soft bits, then we apply the transform in (3) to convert them to soft bits. The failed, soft codeword is first split into matrices of soft bits corresponding to different MIMO channel uses, and each matrix Λ is fed to the encoder f_{Q} and quantized, yielding a bit string representing the quantized soft bit matrix. This representation is stored, and the soft bits are decoded (reconstructed) using g_{Q} and converted to log-likelihood using the arctanh function when the second codeword transmission arrives and soft combining is performed.

3 Results and discussion

3.1 Verifying Theorems 1 and 2

To verify the two bounds for ML 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 × 2, 64-QAM, i.i.d. Rayleigh fading scenario, but this result also holds for the 4 × 4, 16-QAM, i.i.d. Rayleigh fading scenario used in Sect. 3.3. For the considered scenario, Theorem 1 states that the latent feature dimension corresponding to the lower bound is eight, while Theorem 2 gives a dimension of six matching the upper bound.

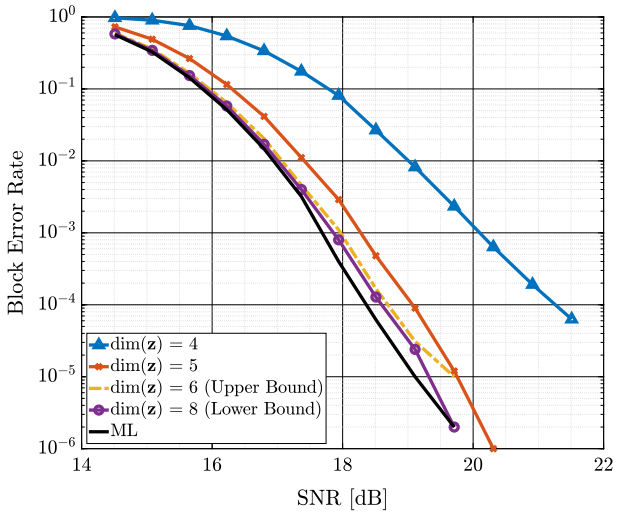

Fig. 5 Impact of latent feature dimension on different instances of EQ-Net without numerical quantization in an i.i.d. Rayleigh fading, 2 × 2, 64-QAM scenario using an (324, 648) LDPC code. A feature dimension of six corresponds to the upper bound in Theorem 2, which is proven explicitly for diagonalized channels, but is shown here to be practically achievable for arbitrary channels

Figure 5 shows the end-to-end block error rate performance averaged over 10000 codewords when the feature dimension is the only parameter that changes in a series of autoencoder models. In all cases, the ML soft bit matrix is estimated using (4), and used to train and evaluate the autoencoder with a given bottleneck size. 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 Fig. 5 is due to the residual training error, even when quantization is not applied to the feature space. Because optimizing the weights of a deep neural network with nonlinear activation is a non-convex problem, the training loss in (18) is not exactly zero at the end of training, or for the test samples. This implies that soft bits will always be distorted by f_{Q} and g_{Q}, even without numerical quantization. In practice, this could be mitigated by training the model for significantly longer, and using significantly more training data, as well as using cyclical learning rate schedules that escape from local minima [48].
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 in error and departure from ML performance, providing evidence that the representation size ratio of Theorem 2 is optimal for the ML soft bits.
3 To capture scenarios that require reliable communications, we simulate SNR values larger than 18.5 dB using one million codewords and still find no significant deviations from optimal performance for $\dim(\mathbf{z}) \geq 6$—note that the error for the $\dim(\mathbf{z}) = 6$ curve is exactly zero at the 20.3 dB point, due to the finite number of simulated codewords. The matching bit error rate in high SNR (greater than 20 dB)

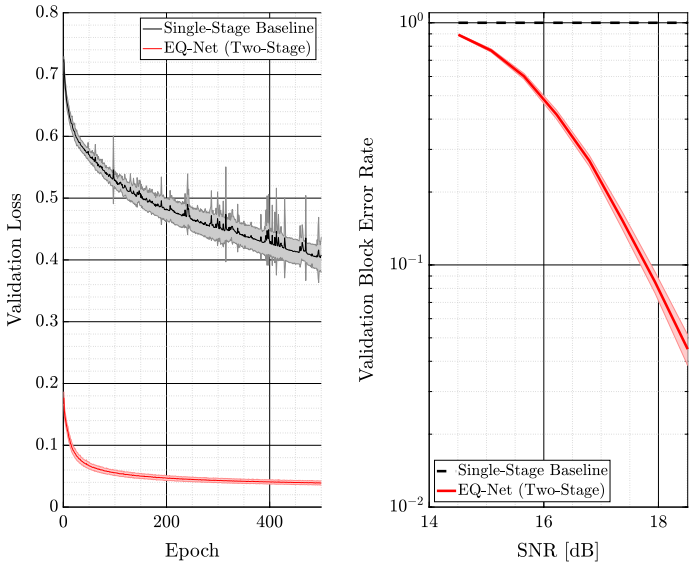

Fig. 6 Comparing the two-stage training of EQ-Net and a naïve supervised baseline that jointly trains the estimation encoder and the decoder from scratch, in a single stage. In both cases, the networks have the same architecture and initial weights, and the same number of gradient updates is performed. The plot includes shaded standard deviation areas over ten runs. (Left) Validation error (soft bit distortion). (Right) Validation coded block error rate with models taken after 500 epochs

is empirically less than 10^{-7}, thus to further reduce this performance gap, an outer code such as the Reed–Solomon code [49], with a high rate, could be used.

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

We investigate the difference in performance of the proposed two-stage learning approach against a single-stage learning baseline that combines (15) and (19). We use exactly the same architecture in both cases. Figure 6 shows the validation loss during training (left), as well as the validation coded block error rate after 500 epochs of training (right), for both methods in a 2×2, 64-QAM, and i.i.d. Rayleigh fading scenario. 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—as shown in the left side of Fig. 6—and converges to a solution with lower coded block error rate—as shown in the right side of Fig. 6. 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.

3.3 Estimation performance

We implement and evaluate two variants of EQ-Net: *EQ-Net (L)* (one residual block) and *EQ-Net (P)* (three residual blocks). We compare our method with two state-of-the-art deep learning baselines: the scheme in [34]—that we further refer to as NN-Det for

Table 1 Latency benchmarks of soft-output MIMO detection algorithms for a 4×4, 16-QAM scenario

	EQ-Net (L)	NN-Det (P)	OAMP-Net2	MMSE-SIC	ZF-SIC	SD
Latency CPU [ms], $B = 1$	1.807	3.456	2.835	0.839	0.760	1.622
Latency CPU [ms], $B = 16$	2.256	3.953	3.227	1.109	1.012	19.52
Latency GPU [ms], $B = 8192$	9.107	17.251	30.501	8.248	6.253	–

B represents the batch size during parallelized inference. All values are expressed in milliseconds and represent average execution times for the entire batch

Table 2 Latency benchmarks of evaluated soft-output MIMO detection algorithms for a 2×2, 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	MMSE-SIC	ZF-SIC	SD
Latency CPU [ms], $B = 1$	1.511	1.903	2.269	2.689	2.125	0.603	0.559	0.6935
Latency CPU [ms], $B = 16$	1.767	2.292	2.565	3.073	2.376	0.831	0.787	4.547
Latency GPU [ms], $B = 8192$	6.523	8.767	12.61	15.91	25.94	7.907	5.311	—

brevity—and the *OAMP-Net2* approach in [32]. We additionally compare with three non-learning-based approaches: ZF-SIC, MMSE-SIC (both implemented via the QR decomposition, as in [39]), and soft-output SD implemented using the default MATLAB algorithm based on [29]. For 4×4, 16-QAM, we train a single NN-Det model with ten unfolded blocks, labeled as high-performance (P). For 2×2, 64-QAM, we train two variants of NN-Det: *NN-Det (P)* (four unfolded blocks), and *NN-Det (L)* three unfolded blocks) which trades off some of the performance for lower end-to-end latency. Implementations of the baselines are available along with our source code.

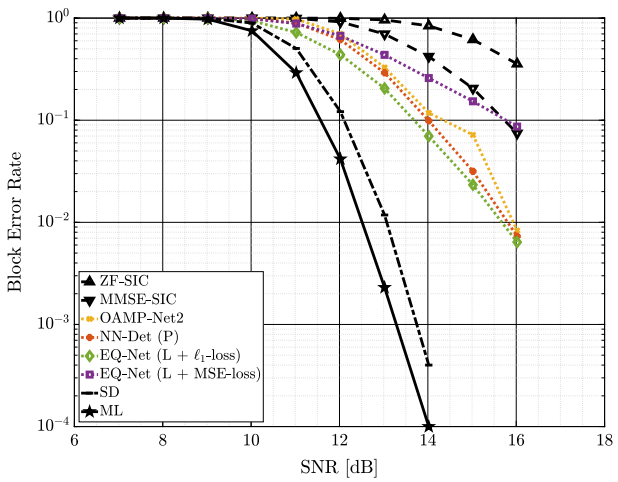

Fig. 7 Estimation (MIMO detection) performance of EQ-Net (including an ablation result on using MSE loss for the second training stage) against state-of-the-art approaches in 4×4, 16-QAM, Rayleigh fading, and LDPC-coded with size (324, 648). Table 1 compares the inference latency of the methods

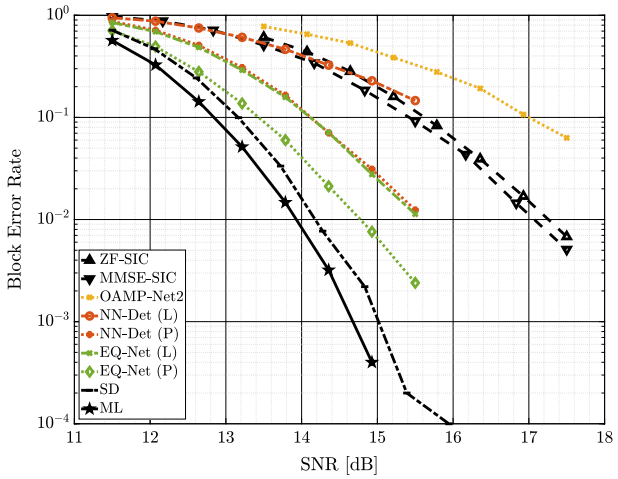

Fig. 8 Estimation (MIMO detection) performance of EQ-Net versus state-of-the-art approaches in 2×2, 64-QAM Rayleigh fading, and LDPC-coded with size (324, 648). Table 2 compares the inference latency of the methods

Figures 7 and 8 plot the performance in 4×4, 16-QAM and 2×2, 64-QAM scenarios, respectively. For the 64-QAM scenario, EQ-Net (L) achieves the same performance as the high-performance NN-Det (P). This is better highlighted in Fig. 7, where the same conclusion is evident. We observe that using the ℓ_1-loss in (19) surpasses the MSE version. We attribute this to the fact that the feature representation is bounded to the $(-1, 1)$ interval in the first training stage, causing the MSE to shrink—and slow down learning—when squared errors are used in the second stage. In contrast, the ℓ_1-loss efficiently learns when features are concentrated around the origin. The performance of EQ-Net also surpasses the ZF-SIC and MMSE-SIC baselines, but there is still a gap with respect to the SD algorithm.

We quantify end-to-end latency for processing a batch of B samples (including transfer to/from GPU where applicable) by measuring the average (using 10,000 batches) execution times using the `timeit` Python module for all algorithms. For the CPU measurements, the processing is executed on a single thread. The CPU is an Intel i9-9900x running at 3.5 GHz, and the GPU is an NVIDIA RTX 2080Ti. From Table 2 and Table 1, it is observed that, while SD is efficient for a batch size of one, the degree of parallelism is much lower, explained by the heavy use of sorting and the tree search procedure [29]. In contrast, the deep learning-based approaches do not require sorting operations, leading to more efficient parallel implementations that favorably scale with the batch size. ZF-SIC and MMSE-SIC also show low latency in both scenarios, but suffer from a performance drop. 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) and NN-Det (P) use a total of 195k and 260k, respectively. 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).

The reduced latency compared to the other deep learning methods is attributed to the fact that the baselines rely on unfolded detection approaches and require additional linear algebra computations, such as matrix inversion and conjugate multiply

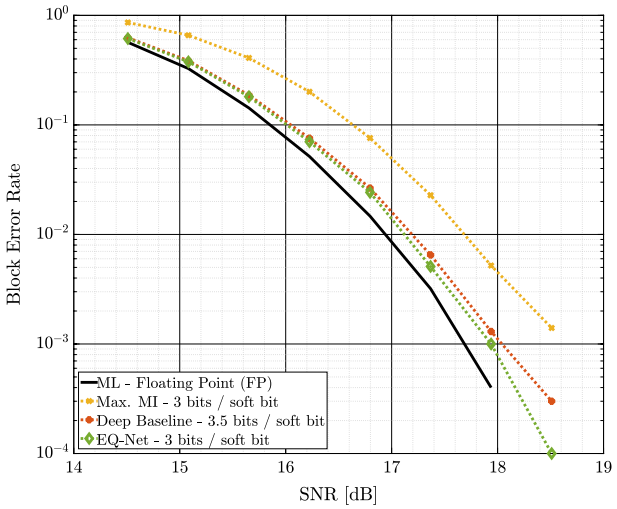

Fig. 9 Quantization performance of EQ-Net evaluated against state-of-the-art methods in 2 × 2, 64-QAM, i.i.d. Rayleigh fading, and LDPC-coded with size (324, 648)

after each iteration. These operations are sequential and increase the end-to-end inference latency. In contrast, our approach does not involve any additional operations. The last lines in Tables 1 and 2 highlight the increased latency gap when the algorithm is used in a scenario with a large number of orthogonal (e.g., multiplexed in the frequency domain) SU-MIMO users, where the base station performs computations in large batches. In this case, EQ-Net is competitive in terms of latency with the ZF-SIC and MMSE-SIC baselines.

3.4 Quantization performance

For the quantization task, we investigate the performance of EQ-Net against the maximum mutual information (MI) quantizer in [20] and the deep learning baseline in [24], both designed for SISO scenarios. We extend these methods to MIMO scenarios by splitting the soft bit matrix into rows and quantizing each of them separately. To test the quantization methods, random payloads are generated, encoded using an LDPC code of size (324, 648), modulated using QAM, transmitted across a number of different MIMO channels (with fading and noise). At the receiver, soft bits are first estimated with either the ML, MMSE-SIC, or LMMSE algorithms, after which finally the soft bits are quantized and reconstructed to evaluate distortions. For EQ-Net, 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 feature, per MIMO channel use. For a 2 × 2, 64-QAM scenario, this amounts to compressing the soft bit matrix (twelve soft bits per MIMO channel use) using a 36-bit codeword, leading to an effective storage cost of three bits per soft bit.

The results in Fig. 9 show that EQ-Net is superior to both baselines and can efficiently quantize the soft bit matrix, with a minimal performance loss. Compared to the deep learning baseline, EQ-Net achieves a 16% compression gain at the same end-to-end performance, whereas compared to the maximum MI quantizer, EQ-Net

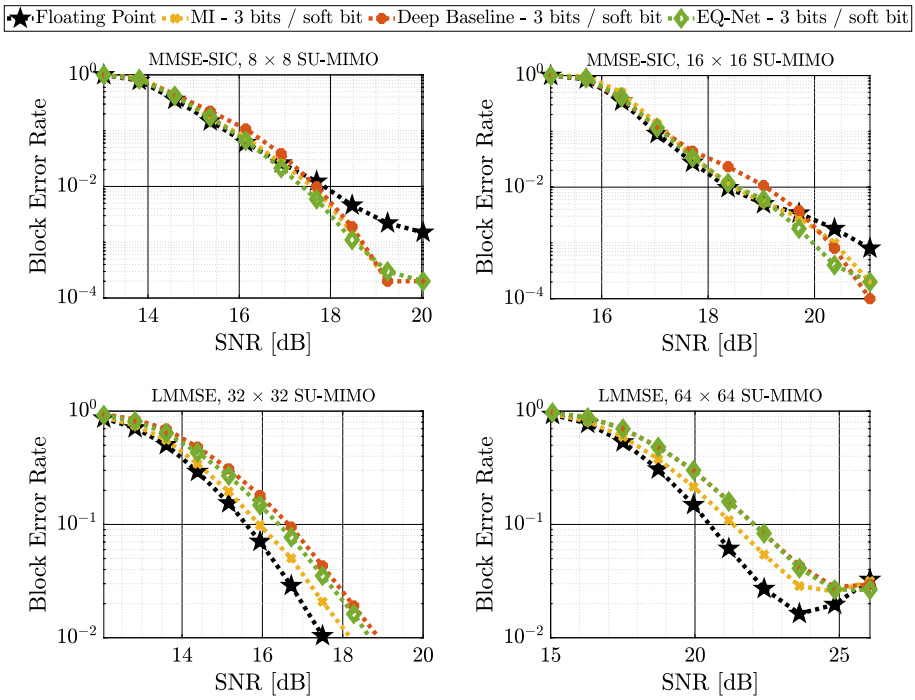

Fig. 10 Quantization performance of EQ-Net evaluated against state-of-the-art methods in large and very large MIMO, 16-QAM, i.i.d. Rayleigh fading, and LDPC-coded with size (324, 648). **a** 8 × 8 MIMO, using MMSE-SIC soft bits for training and testing. **b** 16 × 16 MIMO, using MMSE-SIC soft bits for training and testing. **c** 32 × 32 MIMO, using MMSE-SIC soft bits for training and LMMSE soft bits for testing. **d** 64 × 64 MIMO, using MMSE-SIC soft bits for training and LMMSE soft bits for testing

boosts the performance of the system by 0.65 dB while using the same storage budget. This increase in performance highlights the importance of jointly learning a feature space across the spatial dimensions of MIMO channels.

3.5 Quantization in large-scale MIMO scenarios

When considering large, spatial multiplexing SU-MIMO scenarios with a high modulation order, the ML estimate $\Lambda^{(\mathrm{ML})}$ is no longer practically attainable even during the offline training stage. Note that, according to Theorem 2, a modulation order of at least $K = 4$ (16-QAM) is required to achieve feature compression, which justifies our choice of $K = 4$ for the remainder of this section.

For 8 × 8 and 16 × 16 MIMO scenarios with 16-QAM, Fig. 10a and b shows the performance of EQ-Net and the two quantization baselines, when three bits are used per soft bit. It can be seen that EQ-Net surpasses both baselines at the same bit budget. More interestingly, all quantization methods *improve* the performance compared to floating point soft bit estimation with MMSE-SIC. We attribute this effect to an effect similar to that which occurs in systems that use hybrid hard/soft bit estimation [50]—clipping log-likelihood ratios (in our method, implicitly, by quantizing the reconstructed soft bits $\tilde{\Lambda}$) using well-tuned clipping levels can compensate for hard errors introduced in the MMSE-SIC detection chain, especially for larger MIMO scenarios.

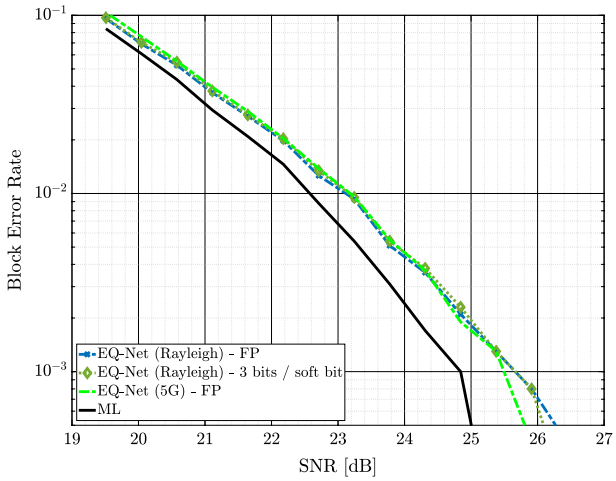

Fig. 11 Quantization performance of EQ-Net under a severe distributional shift induced by training on i.i.d. Rayleigh fading channels and testing on the 5G-NR CDL-A MIMO channel model for 2 × 2, 64-QAM, and LDPC-coded with size (324, 648)

Figure 10c and d plots the performance of EQ-Net and the baselines, when quantizing LMMSE soft bits in 32 × 32 and 64 × 64 scenarios with 16-QAM, respectively. In the case of EQ-Net, we do not train new models f_{Q} and g_{Q}, but instead reuse the models trained for 16 × 16, 16-QAM by separately quantizing sub-matrices of 16 rows from the soft bit matrix $\Lambda^{(\mathrm{LMMSE})}$. Empirically, we find that learning a model for 32 × 32 from scratch involves very wide networks that are difficult and costly to train, as width scales with N_{t}, according to Fig. 2.

From Fig. 10c and d, we conclude that the quantization part of EQ-Net is robust to input shifts in the quantization mode and can scale to large MIMO scenarios—even when trained on soft bits derived from the MMSE-SIC algorithm, the model can quantize LMMSE soft bits, surpassing the deep baseline, and remaining competitive with the MI baseline.

3.6 Robustness to distributional shifts

Practical communication scenarios involve test-time distributional shifts, or may be faced with imperfect CSI during deployment. For soft bit estimation and quantization, the most fragile part of the system is given by the matrix **H**, and any distributional mismatch that occurs. To evaluate the robustness of our approach, we consider models trained on channels from an i.i.d. Rayleigh fading model and tested on realizations of the CDL-A channel model adopted by the 5G-NR specifications [51].

Figure 11 shows the quantization performance under this shift in a 2-by-2 64-QAM scenario, where *EQ-Net (Rayleigh)* is a model trained using Rayleigh channels and *EQ-Net (5G)* is an identical model trained using 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 (Rayleigh) model is robust and does not exhibit error floors or performance losses. This is a strong indicator that the quantization learned using Rayleigh channels can be used in a wide variety of conditions and is further discussed in the section.

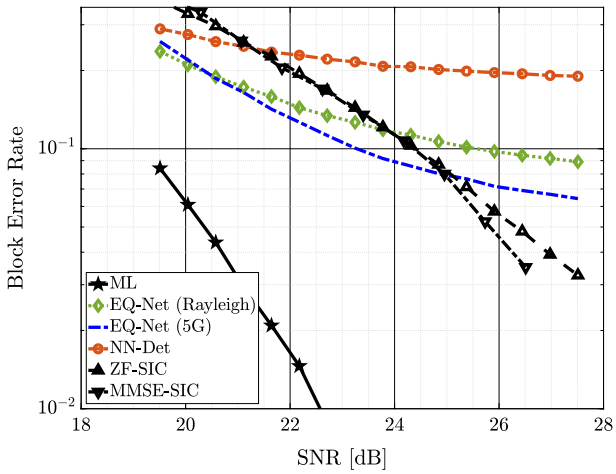

Fig. 12 Estimation performance of EQ-Net (L), and the similarly performing NN-Det (P) under a severe distributional shift induced by training on an i.i.d. Rayleigh fading channel, and testing on the 5G-NR CDL-A MIMO channel model for 2×2, 64-QAM, and LDPC-coded with size (324, 648)

Figure 12 investigates the estimation performance under the same shift and reveals a higher degree of robustness compared to the NN-Det approach, retaining a performance close to that of the ZF-SIC and MMSE-SIC algorithms in the low- and mid-SNR regimes, and being surpassed in the high SNR regime. We attribute this error floor to the ill-conditioned channels in the CDL-A models: Because we are using spatial multiplexing in this scenario, all sub-optimal algorithms suffer a performance drop, and the shallow EQ-Net is no longer sufficient to achieve near-ML performance. A potential fix here is to increase the depth of the estimation encoder f_{E}, at the cost of increased latency.

We finally 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 $\sigma^2_{\mathrm{CSI}}\mathbf{I}_{N_{\mathrm{r}} \times N_{\mathrm{t}}}$. This models impairments coming from the channel estimation module and is a widely used model to test the robustness of downstream estimation algorithms [52, 53]. Figure 4 plots test performance with corrupted CSI in a 2×2 MIMO scenario with 64-QAM and shows that EQ-Net is as robust as the NN-Det baseline to this type of impairment, while still benefiting from the lower latency in Table 2.

3.7 Limitations: estimation in large-scale MIMO scenarios

We have attempted to train the estimation part (second stage) of EQ-Net for large MIMO scenarios and found that this is faced in the following challenges:

- Because the width of the estimation encoder is proportional to N_{t}^2, the total number of weights increases in the order of $\mathcal{O}(N_{\mathrm{t}}^4)$, becoming prohibitive even for $N_{\mathrm{t}} = 8$ (more than three million weights).
- Because we can only generate MMSE-SIC soft bits for training, learning to estimate them is difficult when using a feed-forward architecture, due to the recursive nature

of the algorithm. Using a recurrent architecture for f_{E} could be a potential fix for this issue, but would involve major changes to the design in Fig. 2(c).

- Finally, the latency of MMSE-SIC is still acceptable in 8×8 and 16×16 scenarios (e.g., below 5 ms for a batch size of one, on CPU)—thus, supervised learning of soft bit estimation with deep learning is faced with the challenge of learning from sub-optimal estimates, and it is still an open problem if deep learning can significantly surpass the algorithm used to generate the training soft bits in performance.

Taken together, the above issues currently limit the estimation part of the EQ-Net framework to be advantageous against competing methods only in small MIMO scenarios, up to 4×4, where the ML soft bits (which do not recurrent evaluation) are tractable during the offline training stage. A promising direction of future research is to incorporate algorithm specific changes—such as the recurrent nature of MMSE-SIC—into the deep learning architecture of f_{E}, while still using the two-stage approach of EQ-Net. A recent example of architectural design in this sense is given in [54], where interference cancelation steps are interleaved with a learnable model.

Finally, a large portion of the complexity of f_{E} comes from taking as input the full CSI matrix $\mathbf{H}$. Without further assumptions on the distribution of $\mathbf{H}$ in an environment, this is required for accurate soft bit estimation. Using only incoherent information about $\mathbf{H}$, obtained offline, could be a way of significantly reducing inference complexity in environments with strong assumptions.

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 distortion-free representations of ML and MMSE-SIC soft bits in MIMO channels and further showed evidence that the upper bound for ML soft bits 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 drawback that remains is the presence of an error floor when faced with severe distributional shifts at test time, as per Fig. 12. Even though our results show that this floor is much lower than that of the prior deep learning-based work in [34], there is still room for improvement, at least in overcoming the MMSE-SIC algorithm across the entire SNR range. For example, our method could 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 architectures to learn universal algorithms for soft bit processing.

Appendix

Proof of Theorem 1

Let the QR decomposition of $\mathbf{H}$ be given by $\mathbf{H} = \mathbf{QR}$, where $\mathbf{Q}$ satisfies $\mathbf{Q}^{\mathrm{H}}\mathbf{Q} = \mathbf{I}_{N_\mathrm{t}}$ 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 $\mathbf{Q}^{\mathrm{H}}$, the system in (1) becomes

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

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

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

Thus, there exists a surjective function that maps the features $\tilde{\mathbf{y}}/\sigma_n^2$ and $\mathbf{R}/\sigma_n^2$ to the matrix of soft bits $\Lambda^{(\mathrm{ML})}$ 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. As $\tilde{\mathbf{y}}$ is a complex-valued vector of length N_t, it can be stored using $2N_\mathrm{t}$ real-valued features. $\mathbf{R}$ is a complex-valued upper triangular matrix of size $N_\mathrm{t} \times N_\mathrm{t}$ with real-valued elements on its diagonal, and it can be stored using N_t and $2\frac{(N_\mathrm{t}-1)N_\mathrm{t}}{2}$ real-valued features for its diagonal and off-diagonal terms, respectively. Thus, we have that

$$2N_\mathrm{t} + N_\mathrm{t} + 2\frac{(N_\mathrm{t}-1)N_\mathrm{t}}{2} = N_\mathrm{t}(N_\mathrm{t}+2)$$

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

Proof of Theorem 2

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

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

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} = \mathbf{USV}^{\mathrm{H}}\mathbf{V}\mathbf{x} + \mathbf{n} = \mathbf{USx} + \mathbf{n}.$$

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

$$\tilde{\mathbf{y}} = \mathbf{U}^{\mathrm{H}}\mathbf{USx} + \mathbf{U}^{\mathrm{H}}\mathbf{n} = \mathbf{Sx} + \tilde{\mathbf{n}},$$

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

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

According to [24], 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 $\Lambda^{(\mathrm{ML})}$ can be represented using a $3N_\mathrm{t}$-dimensional feature representation when the channel is diagonalized. The representation size ratio of this function is equal to $R_{\mathrm{up,ML}} = \frac{KN_\mathrm{t}}{3N_\mathrm{t}} = \frac{K}{3}$.

Given that $N_\mathrm{t} \geq 1$, we have that R_{up} represents an upper representation size ratio bound for $K \geq 3$ and arbitrary channels $\mathbf{H}$, because at least the subset of soft bits derived from diagonalized channels cannot be represented using fewer features without further assumptions.

Proof of Theorem 3

The proof is by induction. Considering (12) and assuming that n_i is Gaussian, then $\tilde{n}_i$ is also Gaussian, as $\mathbf{n}$ and $\mathbf{Q}$ are independent. The closed-form expression of the kth LLR of the N_tth symbol is given by

$$L_{k,N_\mathrm{t}}^{(\mathrm{MMSE-SIC})} = \log \frac{\sum_{x_j, b_k=1} \exp\left(-\frac{|\tilde{y}_{N_\mathrm{t}} - \underline{r}_{N_\mathrm{t},N_\mathrm{t}} x_j|^2}{\sigma^2_{\tilde{n}_{N_\mathrm{t}}}}\right)}{\sum_{x_j, b_k=0} \exp\left(-\frac{|\tilde{y}_{N_\mathrm{t}} - \underline{r}_{N_\mathrm{t},N_\mathrm{t}} x_j|^2}{\sigma^2_{\tilde{n}_{N_\mathrm{t}}}}\right)}.$$

Then, the above equation is a function $f_{N_\mathrm{t}}(\tilde{y}_{N_\mathrm{t}}/\sigma^2_{\tilde{n}_{N_\mathrm{t}}}, \underline{r}_{\mathrm{t,t}}/\sigma^2_{\tilde{n}_{N_\mathrm{t}}}) = L_{:,N_\mathrm{t}}^{(\mathrm{MMSE-SIC})}$. Given that $\tilde{y}_{N_\mathrm{t}}$ is a complex scalar, $\underline{r}_{N_\mathrm{t},N_\mathrm{t}}$ is a real scalar, and $\sigma^2_{\tilde{n}_{N_\mathrm{t}}}$ is real scalar, it follows that the LLR vector corresponding to the last spatial stream can be exactly represented by a three-dimensional real vector. This proves the case $N_\mathrm{t} = 1$, where sequential detection is completed.

The equation for estimating the LLR values corresponding to the ith symbol, given the previous $N_\mathrm{t} - i$ estimates given by

$$\tilde{y}_i = \underline{r}_{i,i} x_i + \sum_{j=i+1}^{N_\mathrm{t}} \underline{r}_{i,j} x_j + \tilde{n}_i.$$

Letting $\hat{y}_i = \tilde{y}_i - \sum_{j=i+1}^{N_\mathrm{t}} \underline{r}_{i,j} x_j^{(\mathrm{MMSE-SIC})}$, and treating the remaining interference as Gaussian, we obtain the compact model in (12) as

$$\hat{y}_i = \underline{r}_{i,i} x_i + \hat{n}_i,$$

and there exists a function $f_i(\hat{y}_i/\sigma^2_{\hat{n}_i}, \underline{r}_{\mathrm{i,i}}/\sigma^2_{\hat{n}_i}) = L_{:,i}^{(\mathrm{MMSE-SIC})}$ for each i.

It follows that, given estimates for the previous $N_\mathrm{t} - i$ symbols, the LLR values of the ith symbol can be exactly represented by a vector with three real values, regardless of the modulation order. Thus, by induction, there exists a function $f : \mathbb{R}^{3N_\mathrm{t}} \to \mathbb{R}^{K \times N_\mathrm{t}}$ such that $f(\hat{\mathbf{y}}, \mathrm{diag}(\underline{\mathbf{R}})) = \mathbf{L}^{(\mathrm{MMSE\text{-}SIC})}$. Using that $\Lambda_{i,k} = \tanh \frac{L_{i,k}}{2}$ is a bijective function of L, for all i, k, it follows that $R_{\mathrm{up,MMSE-SIC}} = \frac{KN_\mathrm{t}}{3N_\mathrm{t}} = \frac{K}{3}$.

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
MMSE-SIC	Minimum mean squared error with successive interference cancelation
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 cancelation

Acknowledgements
The authors would like to thank Ian Roberts for providing early feedback for the manuscript. The authors would also like to thank the editor and the anonymous reviewers for their constructive feedback during the review process.

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

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.

Data availability statement
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, is publicly available in the EQ-Net Github repository, located at https://github.com/utcsilab/eq-net.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 28 January 2022 Accepted: 24 May 2022
Published online: 13 June 2022

References

1. H. Viswanathan, P.E. Mogensen, Communications in the 6G era. IEEE Access **8**, 57063–57074 (2020)
2. J. Hagenauer, Soft is better than hard. Communications and Cryptography, 155–171 (1994)
3. B. Zhao, M.C. Valenti, Practical relay networks: a generalization of hybrid-ARQ. IEEE J. Sel. Areas Commun. **23**(1), 7–18 (2005)
4. Y. Song, N. Devroye, Lattice codes for the gaussian relay channel: Decode-and-forward and compress-and-forward. IEEE Trans. Inf. Theory **59**(8), 4927–4948 (2013)
5. T.X. Vu, H.D. Nguyen, T.Q. Quek, Adaptive compression and joint detection for fronthaul uplinks in cloud radio access networks. IEEE Trans. Commun. **63**(11), 4565–4575 (2015)
6. P. Kairouz, H.B. McMahan, B. Avent, A. Bellet, M. Bennis, A.N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings et al, Advances and open problems in federated learning. arXiv preprint arXiv:1912.04977 (2019)
7. S. Feng, D. Niyato, P. Wang, D.I. Kim, Y.-C. Liang, 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
8. P. Frenger, S. Parkvall, E. Dahlman, 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
9. W. Ejaz, G. Hattab, N. Cherif, M. Ibnkahla, F. Abdelkefi, M. Siala, Cooperative spectrum sensing with heterogeneous devices: Hard combining versus soft combining. IEEE Syst. J. **12**(1), 981–992 (2016)
10. E.G. Larsson, P. Stoica, J. Li, On maximum-likelihood detection and decoding for space-time coding systems. IEEE Trans. Signal Process. **50**(4), 937–944 (2002)

11. M. Bennis, M. Debbah, H.V. Poor, Ultrareliable and low-latency wireless communication: Tail, risk, and scale. Proc. IEEE **106**(10), 1834–1853 (2018)
12. 3GPP: 38.214: Physical Layer Procedures for Data. (2021). 3GPP. Ver. 16.4
13. M.A. Albreem, M. Juntti, S. Shahabuddin, Massive MIMO detection techniques: a survey. IEEE Commun. Surv. Tutor. **21**(4), 3109–3132 (2019)
14. C. Zhang, P. Patras, H. Haddadi, Deep learning in mobile and wireless networking: a survey. IEEE Commun. Surv. Tutor. **21**(3), 2224–2287 (2019)
15. T. Erpek, T.J. O'Shea, Y.E Sagduyu, Y. Shi, T.C. Clancy, Deep Learning for Wireless Communications (2020). arXiv:2005.06068
16. N. Akhtar, A. Mian, Threat of adversarial attacks on deep learning in computer vision: a survey. Ieee Access **6**, 14410–14430 (2018)
17. D. Krueger,E. Caballero, J.-H. Jacobsen, A. Zhang, J. Binas, D. Zhang, R. Le Priol, A. Courville, Out-of-distribution generalization via risk extrapolation. In: International Conference on Machine Learning, pp. 5815–5826 (2021). PMLR
18. S. Kokalj-Filipovic, R. Miller, Adversarial examples in RF deep learning: detection of the attack and its physical robustness. arXiv preprint arXiv:1902.06044 (2019)
19. Q. Hu, F. Gao, H. Zhang, S. Jin, G.Y. Li, Deep learning for channel estimation: interpretation, performance, and comparison. IEEE Trans. Wireless Commun. **20**(4), 2398–2412 (2020)
20. A. Winkelbauer, G. Matz, 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)
21. G. Zeitler, G. Bauch, J. Widmer, Quantize-and-forward schemes for the orthogonal multiple-access relay channel. IEEE Trans. Commun. **60**(4), 1148–1158 (2012)
22. M. Danieli, S. Forchhammer, J.D. Andersen, L.P. Christensen, S.S. Christensen, Maximum mutual information vector quantization of log-likelihood ratios for memory efficient HARQ implementations. In: 2010 Data Compression Conference, pp. 30–39 (2010). IEEE
23. W. Rave, Quantization of log-likelihood ratios to maximize mutual information. IEEE Signal Process. Lett. **16**(4), 283–286 (2009)
24. M. Arvinte, A.H Tewfik, S. Vishwanath, Deep log-likelihood ratio quantization. In: 2019 27th European Signal Processing Conference (EUSIPCO), pp. 1–5 (2019). IEEE
25. P.W. Wolniansky, G.J. Foschini, G.D. Golden, R.A. Valenzuela, 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)
26. C. Shen, Y. Zhu, S. Zhou, J. Jiang, 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
27. P. Luethi, A. Burg, S. Haene, D. Perels, N. Felber, W. Fichtner, VLSI implementation of a high-speed iterative sorted MMSE QR decomposition. In: 2007 IEEE International Symposium on Circuits and Systems, pp. 1421–1424 (2007). IEEE
28. B. Hassibi, H. Vikalo, On the sphere-decoding algorithm I. Expected complexity. IEEE Trans. Signal Process. **53**(8), 2806–2818 (2005)
29. C. Studer, A. Burg, H. Bolcskei, Soft-output sphere decoding: Algorithms and VLSI implementation. IEEE JSAC **26**(2), 290–300 (2008)
30. Y. Wu, J. McAllister, Configurable quasi-optimal sphere decoding for scalable MIMO communications. IEEE Trans. Circuits Syst. I Regul. Pap. **68**(6), 2675–2687 (2021)
31. M. Mohammadkarimi, M. Mehrabi, M. Ardakani, Y. Jing, Deep learning-based sphere decoding. IEEE Trans. Wireless Commun. **18**(9), 4368–4378 (2019)
32. H. He, C.-K. Wen, S. Jin, G.Y. Li, Model-driven deep learning for MIMO detection. IEEE Trans. Signal Process. **68**, 1702–1715 (2020)
33. J. Ma, L. Ping, Orthogonal AMP. IEEE Access **5**, 2020–2033 (2017)
34. O. Sholev, H.H. Permuter, E. Ben-Dror, W. Liang, Neural network MIMO detection for coded wireless communication with impairments. In: Proc. IEEE Wireless Commun. Netwk. Conf., pp. 1–8 (2020)
35. O. Shental, J. Hoydis, Machine LLRning: Learning to softly demodulate. In: Proc. IEEE Global Commun. Conf. Workshops, pp. 1–7 (2019)
36. D. Tse, P. Viswanath, *Fundamentals of Wireless Communication* (Cambridge University Press, Cambridge, 2005)
37. J. Chen, A. Dholakia, E. Eleftheriou, M.P. Fossorier, X.-Y. Hu, Reduced-complexity decoding of LDPC codes. IEEE Trans. Commun. **53**(8), 1288–1299 (2005)
38. B. Hanin, 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)
39. R.T. Kobayashi, T. Abrão, Ordered MMSE-SIC via sorted QR decomposition in ill conditioned large-scale MIMO channels. Telecommun. Syst. **63**(2), 335–346 (2016)
40. R.W. Heath, N. Gonzalez-Prelcic, S. Rangan, W. Roh, A.M. Sayeed, An overview of signal processing techniques for millimeter wave MIMO systems. IEEE J. Select. Top. Signal Process. **10**(3), 436–453 (2016)
41. H. Sampath, P. Stoica, A. Paulraj, Generalized linear precoder and decoder design for MIMO channels using the weighted MMSE criterion. IEEE Trans. Commun. **49**(12), 2198–2206 (2001)
42. A. Kipnis, G. Reeves, Gaussian approximation of quantization error for estimation from compressed data. IEEE Trans. Inf. Theory **67**(8), 5562–5579 (2021)
43. D. Arthur, S. Vassilvitskii, k-means++ the advantages of careful seeding. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1027–1035 (2007)
44. D.P. Kingma, J. Ba, Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
45. M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *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)
46. M. Biguesh, A.B. Gershman, Training-based MIMO channel estimation: a study of estimator tradeoffs and optimal training signals. IEEE Trans. Signal Process. **54**(3), 884–893 (2006)

47. Maini, P., Wong, E., Kolter, Z.: Adversarial robustness against the union of multiple perturbation models. In: International Conference on Machine Learning, pp. 6640–6650 (2020). PMLR
48. Smith, L.N.: A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. arXiv preprint arXiv:1803.09820 (2018)
49. C. Thommesen, The existence of binary linear concatenated codes with reed-solomon outer codes which asymptotically meet the Gilbert-Varshamov bound. IEEE Trans. Inf. Theory **29**(6), 850–853 (1983)
50. Zimmermann, E., Milliner, D.L., Barry, J.R., Fettweis, G.: Optimal LLR clipping levels for mixed hard/soft output detection. In: IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference, pp. 1–5 (2008). IEEE
51. 3GPP: Study on Channel Model for Frequency Spectrum Above 6 GHz. (2019). 3GPP
52. R. Narasimhan, Error propagation analysis of V-BLAST with channel-estimation errors. IEEE Trans. Commun. **53**(1), 27–31 (2005)
53. N.I. Miridakis, T.A. Tsiftsis, On the joint impact of hardware impairments and imperfect CSI on successive decoding. IEEE Trans. Veh. Technol. **66**(6), 4810–4822 (2016)
54. N. Shlezinger, R. Fu, Y.C. Eldar, DeepSIC: Deep soft interference cancellation for multiuser MIMO detection. IEEE Trans. Wireless Commun. **20**(2), 1349–1362 (2020)

Publisher's Note