Universal Auto-encoder Framework for MIMO CSI Feedback

Jinhyun So
Samsung Semiconductor Inc.
San Diego, USA
jinhyun.so@samsung.com

Hyukjoon Kwon
Samsung Semiconductor Inc.
San Diego, USA
hyukjoon.k@samsung.com

***Abstract*—Existing auto-encoder (AE)-based channel state information (CSI) frameworks have focused on a specific configuration of user equipment (UE) and base station (BS), and thus the input and output sizes of the AE are fixed. However, in the real-world scenario, the input and output sizes may vary depending on the number of antennas of the BS and UE and the allocated resource block in the frequency dimension. A naive approach to support the different input and output sizes is to use multiple AE models, which is impractical for the UE due to the limited HW resources. In this paper, we propose a universal AE framework that can support different input sizes and multiple compression ratios. The proposed AE framework significantly reduces the HW complexity while providing comparable performance in terms of compression ratio-distortion trade-off compared to the naive and state-of-the-art approaches.**

***Index Terms*—MIMO, CSI feedback, auto-encoder, various input size, multiple compression ratios**

I. Introduction

In a MIMO system, real-time channel state information (CSI) at the base station (BS) plays an important role in taking advantage of advanced multi-input multi-output (MIMO) techniques. However, in frequency division duplex (FDD) systems, the user equipment (UE) needs to estimate the downlink CSI based on reference signals and send it back to the BS. The communication overhead for CSI feedback becomes one of the major challenges for Massive MIMO in FDD systems, which presents a non-trivial trade-off between CSI distortion and feedback rate. To address this challenge, several methods using compressed sensing (CS) and codebooks have been introduced and applied to LTE and New Radio (5G). The complexity of codebook design and exploiting the sparsity of CSI increases exponentially with the number of transmit and receive antennas, making it impractical in MIMO systems.

Recently, machine learning (ML)-based methods for CSI compression have been studied [1]–[4]. Most of the works in this line consider an auto-encoder (AE) architecture consisting of the pair of encoder and decoder, where the encoder takes CSI as input and outputs a latent vector whose size is much smaller than the size of the input. The decoder in AE aims to reconstruct the original CSI based on the latent vector. The goal of AE is to learn a nonlinear manifold of the CSI with a few dimensions, which is well suited to the CSI feedback problem, as shown in Fig. 1. It has been shown that the AE-based methods can provide a better trade-off between distortion and feedback rate than the conventional CS and codebook-based schemes [1]–[4].

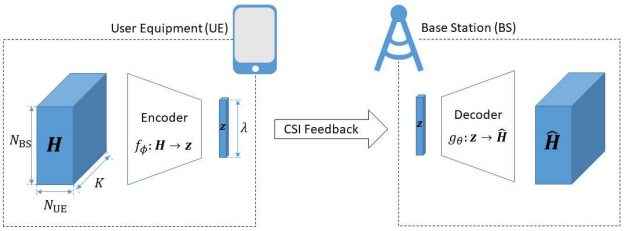


Fig. 1: Auto-encoder (AE) based framework for channel state information (CSI) feedback system.

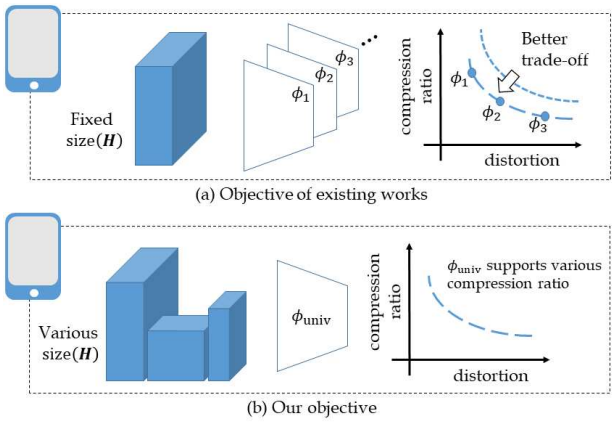


Fig. 2: Comparison of objectives of existing works and ours. (a) Existing works aim to design multiple encoders ($\phi_1, \phi_2, \ldots$) providing a better trade-off between compensation ratio and distortion while the input size is fixed. (b) Our work aims to design a universal encoder ($\phi_{\texttt{univ}}$) which can support various input and latent size to reduce the hardware (HW) complexity of the UE while it shows comparable performance in terms of compression-distortion trade-off.

Most of the existing works have focused on designing an efficient AE framework given the specific configuration, where the number of antennas and the resource allocation to the UE in the frequency domain are fixed. However, in the real-world scenario, the number of transmit and receive antennas can vary among BSs and UEs, and the BS dynamically allocates the resource block in the frequency domain to the UE according to the channel quality. Therefore, the input

dimension of the encoder may change, and thus the UE may need to use multiple encoders to support the different input sizes. In addition, the UE may also need to support different latent sizes and compression ratios (CRs), as the recent work has shown that further communication overhead reduction in CSI feedback can be achieved by adjusting the latent size according to the channel delay profile [5]. The straightforward way to support the different input and latent sizes is to design multiple AE models, each dedicated to the input-latent-size pair. However, due to the limited hardware (HW) resources in mobile devices, implementing multiple encoders for the UE may be impractical.

To address this challenge, our objective is to design a universal AE-based CSI feedback framework that can support different input and latent sizes. Fig. 2 shows the comparison of the objectives of the existing works and ours. Our proposed framework has two outstanding features:

- The encoder framework in the UE is agnostic to the BS configurations such as the number of BS antennas and allocated resource block from the BS.
- A single set of ML parameters of the encoder can support multiple compression ratios (CRs). The encoder contains a universal block without additional layers for each CR, and we propose a novel ML training scheme using a masking layer, which makes the output of the universal block contain more important information in the earlier position.

II. System Model

In this section, we introduce the limited CSI feedback systems for Massive MIMO and provide a standard AE architecture. The AE captures the salient features of high-dimensional MIMO channels, which significantly reduces the feedback overhead. We consider a massive MIMO orthogonal frequency division multiplexing (OFDM) system where a single user (UE) and a single BS have N_{UE} and N_{BS}, respectively. The BS transmits OFDM transmission with N_s data streams over K subcarriers. The received signal on the kth subcarrier can be expressed as

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

where $\mathbf{H}_k \in \mathbb{C}^{N_{BS} \times N_{UE}}$, $\mathbf{V}_k \in \mathbb{C}^{N_{BS} \times N_s}$, $\mathbf{x}_k \in \mathbb{C}^{N_s}$, and $\mathbf{n}_k \in \mathbb{C}^{N_{UE}}$ denotes the channel matrix in frequency domain, precoding matrix at the BS, downlink transmitted data symbol, and additive white Gaussian noise on the k-th subcarrier, respectively. Let $\mathbf{H} = \{\text{real}(\{\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_K\}), \text{imag}(\{\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_K\})\} \in \mathbb{R}^{2 \times K \times N_{BS} \times N_{UE}}$ be a tensor representing the entire CSI stacked by the channel matrices on all subcarriers while real$(\cdot)$ and imag$(\cdot)$ denote the real and imaginary part of the input, respectively. We note that resolution in the frequency dimension can vary according to the granularity of AI-based CSI feedback which can be defined by the network, and in this paper we use a resource block (RB).

Based on the CSI information $\mathbf{H}$, the BS designs the precoding matrixes to improve the spectral efficiency with various MIMO techniques including eliminating the inter-user interference or beamforming. For doing so, the UE estimates and sends $\mathbf{H}$ to the BS, but the total number of feedback parameters in FDD massive MIMO system is linearly increasing with respect to N_{UE},N_{BS}, and K, which is too large in feedback links. To reduce the feedback overhead, the UE can extract the most salient features of the CSI information H by utilizing the AE which employs the pair of encoder and decoder to compress and reconstruct the CSI, respectively. The encoder carries out the compression as

$$\mathbf{z} = f_\phi(\mathbf{H}), \tag{2}$$

where $\mathbf{z} \in \mathbb{R}^\lambda$ is a latent vector of size λ and $f_\phi(\cdot)$ denotes the compression function with parameters ϕ. Then the compression ratio is defined as the ratio between the input and output dimension of the encoder function f_ϕ, which can be expressed by

$$CR_{f_\phi} \triangleq \frac{\texttt{size}(\text{output of } f_\phi)}{\texttt{size}(\text{input of } f_\phi)} = \frac{\lambda}{2KN_{UE}N_{BS}}, \tag{3}$$

where $\texttt{size}(\cdot)$ denotes the number of elements. Note that in this paper, we assume the unlimited representations of continuous features over the latent space. We remain joint optimization of the encoder and quantization to enable discrete representation of the latent space as one of future directions. The UE sends the compressed version of CSI, the latent vector $\mathbf{z}$, to the BS, which significantly reduces the feedback overhead when $\lambda \ll 2KN_{UE}N_{BS}$. Upon receiving $\mathbf{z}$, the BS reconstruct the CSI information by carrying out the decoder, which can be expressed by

$$\hat{\mathbf{H}} = g_\theta(\mathbf{z}), \tag{4}$$

where $g_\theta(\cdot)$ denotes the reconstruction function with a set of parameters θ, and $\hat{\mathbf{H}}$ is the reconstructed CSI tensor which have the same dimensionality as $\mathbf{H}$.

As described in the introduction, CSI tensor $\mathbf{H}$ can have various dimensionality according to the UE and BS antenna configuration, and resource allocation over the frequency dimension. Various feedback overhead (and hence the latent size λ) can be configured to optimize the trade-off between distortion and communication overhead [5]. Let $\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_{max}\}$ be a set of latent sizes that we need to support and λ_{max} is the maximum value of the latent vector size. Let $\mathcal{D} = \{D_1, D_2, \dots\}$ be a set of distributions of CSI tensor, and each distribution may have different size. Straightforward approach to support all pairs in $\Lambda \times \mathcal{D}$ is that for each pair of $(\lambda_i, D_j) \in \Lambda \times \mathcal{D}$, the dedicated pair of encoder and decoder is trained to minimize the reconstruction loss, which can be expressed by

$$\phi^{opt}_{\lambda_i, D_j}, \theta^{opt}_{\lambda_i, D_j} = \arg\max_{\phi, \theta} \mathbb{E}_{\mathbf{H} \sim D_j} \left[L\left(\mathbf{H}, g_{\theta_{\lambda_i}}\left(f_{\phi_{\lambda_i}}(\mathbf{H})\right)\right)\right], \tag{5}$$

where $L(\cdot)$ denotes the loss function. In the worst case, the number of set of the AE pairs becomes $|D| \times |\lambda|$, which can be impractical to be implemented in the UE. To address this challenge, our objective is to design a universal encoder in

the UE which minimizes the total loss function, which can be expressed by

$$\phi^{opt}_{\texttt{univ}}, \{\theta^{opt}_{\lambda_i}\}_{\lambda\in\Lambda} \quad (6)$$
$$= \arg\max_{\phi,\{\theta_\lambda\}_{\lambda\in\Lambda}} \sum_{D\in\mathcal{D}} \sum_{\lambda\in\Lambda} \mathbb{E}_{\mathbf{H}\sim D}\left[L\left(p_D(\mathbf{H}), g_{\theta_\lambda}\left(f_\phi(p_D(\mathbf{H}))\right)\right)\right],$$

where $p_D(\cdot)$ is a pre-processing function to make the output belong to the same space $\mathcal{H}$ while the input size of function p_D depends on D. The role of p_D is to make the input of $\phi_{\texttt{univ}}$ have the same size. Again, our goal is to find the optimal universal encoder $f_\phi : \mathcal{H} \to \mathbb{R}^{\lambda_{\max}}$ from (6) and p_D such that the AE framework can support arbitrary pair of the input and latent size $(\texttt{size}(D), \lambda)$ for all $D \in \mathcal{D}, \lambda \in \Lambda$. The BS may have the different decoder model g_{θ_λ} for each $\lambda \in \Lambda$.

III. Universal Auto-encoder Framework

In this section, we investigate how to design AE-based CSI feedback framework which can support various configurations (i.e., various size of the CSI tensor) and multiple compression ratios (i.e., various size of the latent vector) with the limited HW resource of the UE.

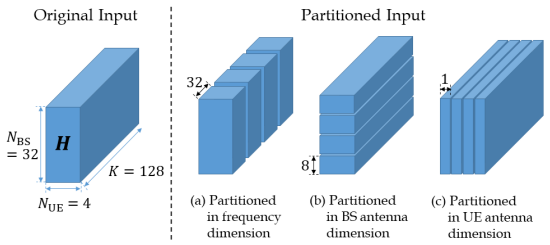
A. Input Space Generalization

At a high-level, input space generalization is composed of two components, 1) partition of input CSI tensor over antenna domains and 2) zero-padding in frequency domain. Partitioned and zero-padded inputs are compressed by a single encoder. The advantage of this partition-based structure is two-folds:

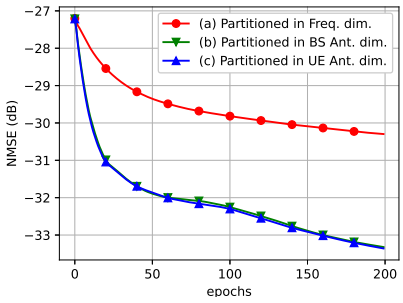
- (Supporting arbitrary input size) As the single (and same) encoder is applied to each antenna element, it can support arbitrary number of UE and BS antennas,
- (Reducing HW complexity) As the input size of the encoder is reduced from the partition (i.e., $\mathbf{H} \in \mathbb{R}^{2\times K\times N_{BS}\times N_{UE}} \to \mathbf{h}^{\text{part}} \in \mathbb{R}^{2\times K\times 1\times 1}$), the number of learnable parameters in the encoder can be significantly reduced. The details will be explained later, but note that the number of parameters of a simple multiple layer perceptron (MLP) structure is reduced from 24 millions (M) to 37 thousands (K).

Then, the natural question is that as the reduced size of encoder is repeatedly applied to different antenna element, what is the performance degradation from the partitioning? Surprisingly, the performance degradation in terms of BLER is negligible while the number of parameters is reduced from 24M to 37K. Now, we will explain the details of the input space generalization.

Compressing the entire CSI tensor $\mathbf{H} \in \mathbb{R}^{2\times K\times N_{BS}\times N_{UE}}$ at once may be intractable to the UE. Let's consider the case where the values of (N_{BS}, N_{UE}, K) is up to $(32, 4, 273)$. The problem is that with these maximum values of input size of the AE, the number of parameters of a simple MLP encoder with one hidden layer, which is one of the simplest ML models, is more than 24M. Due to the limited HW resource of the UE, implementing these parameters is not practical. Given the fixed compression ratio, the number of parameters of the AE is quadratic to the input size in general. Therefore, in this subsection, we investigate how to reduce the input size while preserving the reconstruction performance as much as possible.



(a) Three settings of partitioning $\mathbf{H}$.



(b) NMSE performance of three settings.

Fig. 3: Three settings of partitioning $\mathbf{H}$ to reduce the input size of the AE and their NMSE performance.

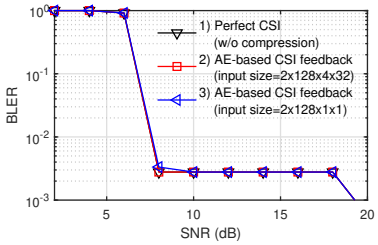


Fig. 4: BLER performance of SVD-based beamforming with three types of CSI feedback.

To reduce the input dimension of the AE, we consider a partition-based approach where the UE partitions $\mathbf{H}$ into multiple parts, each part is compressed by carrying out the encoder, and the UE sends the concatenation of the compressed parts to the BS. Partition can be performed according to the one of dimensions of $\mathbf{H}$, i.e., frequency, BS antenna and UE antenna dimension. To investigate which dimension is *good* to be partitioned, we compare the NMSE performance of three settings that $\mathbf{H} \in \mathbb{R}^{2\times K(=128)\times N_{BS}(=32)\times N_{UE}(=4)}$ is partitioned input 4 parts with respect to frequency, BS antenna, and UE antenna dimension, respectively. Fig. 3a shows the size of the AE input in three settings. *Good* partition means that the NMSE performance does not degrade even if the same encoder is applied to all parts to save the HW resources. Fig. 3b shows the NMSE performance of the three settings of partitioning. We observe that the UE antenna dimension and the BS antenna dimensions are more robust

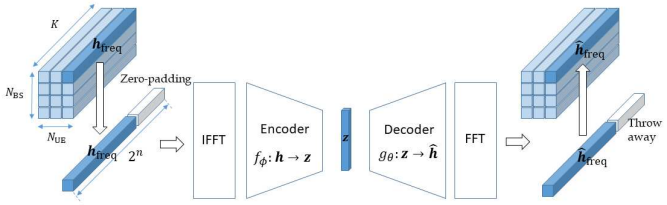


Fig. 5: AE-based CSI feedback framework with the input space generalization when $K \in (2^{n-1}, 2^n]$ (n is an integer).

against performance degradation from partitioning than the frequency dimension. This is because elements in frequency dimension are highly correlated than the other dimensions and the sparsity property over the frequency dimension is good to be compressed.

To further reduce the input size, we investigate how much the block error rate (BLER) performance is degraded when $\mathbf{H}$ is partitioned such that each part has a single element in the BS and UE antenna dimensions, i.e., $\mathbf{h}^{\texttt{part}} \in \mathbb{R}^{2K11}$. Fig. 4 shows the BLER performance of three cases: the BS performs the singular vector decomposition (SVD)-based beamforming with 1) $\mathbf{H}$ (without compression) and 2) reconstructed CSI matrix $\hat{\mathbf{H}} = g_\theta(f_\phi(\mathbf{H}))$ where the CSI tensor is not partitioned, and 3) concatenation of reconstructed channel vectors, i.e., $\hat{\mathbf{H}} = \texttt{concat}\left(\{g_\theta(f_\phi(\mathbf{h}^{\texttt{part}}))\}\right)$. Surprisingly, second and third cases have the almost same BLER performance while the number of parameters are significantly reduced by partitioning the input and applying the same AE model to all parts. Note that the number of parameters of the second and third cases are 24M and 37K, respectively.

Now, the AE can be applied to arbitrary number of elements in the UE and BS antenna dimensions as the CSI tensor can be partitioned such that each part has a single element in those dimension and the same AE model can compress all parts without performance degradation. Remaining one is frequency dimension, whose size depends on CSIRS resource allocation and the number of allocated resource block (RB) can be any integer from 1 to 273. Zero-padding in frequency domain and applying IFFT may enable a single AE to support entire cases of CSIRS resource allocation. In the extreme case, however, to compress a single RB by using the AE whose input size is 273 RB is inefficient in terms of power consumption as 272 elements out of 273 elements in the input of the AE are zero. To avoid this inefficiency, we categorize the input size in frequency dimension into 5 cases according to the number of RB(=K). In each category, $\mathbf{h}^{\texttt{part}}$ is zero-padded such that the number of elements in frequency domain has the form of 2^n where n is an integer, and then is converted to delay dimension by applying 2^n-point IFFT. To save the power consumption, the UE maintains 5 encoders whose input size in delay dimension is 16, 32, 64, 128, and 256, respectively. Table I summarizes the 5 categories with number of RB and IFFT size. Fig. 5 shows the overall framework of the CSI compression and reconstruction.

Now, we can view the set $\mathcal{D}$ in (6) as a set of five distributions $\{D_1, D_2, \ldots, D_5\}$ and D_i is an underlying distribution of the zero-padded and converted (by IFFT) CSI tensor in i-th input category. The precoding function p_{D_i} in (6) is zero-padding to have 2^{i+3} elements and applying 2^{i+3}-point IFFT.

TABLE I: Categories of the input space according to the number of elements in frequency dimension.

Category No.	1	2	3	4	5
RB (=K)	$1 \sim 16$	~ 32	~ 64	~ 128	~ 273
IFFT size	16	32	64	128	256

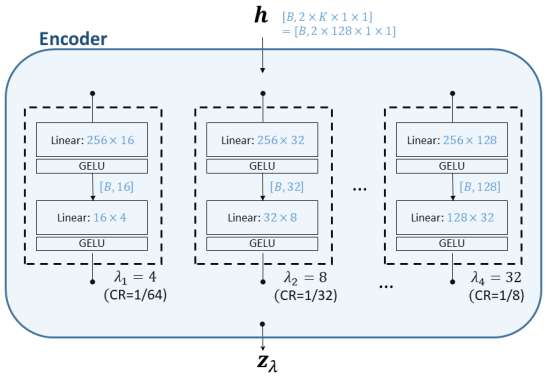
B. Latent Space Generalization

In this subsection, we investigate how to design the AE framework which supports multiple compression ratios. Key intuition behind the latent space generalization is that parameters of the encoder are trained such that earlier position in the latent space contains more important information. Then, the encoder can output the elements starting from the earlier position according to the given compression ratio. Before presenting our proposed approach, we introduce two baseline approaches.

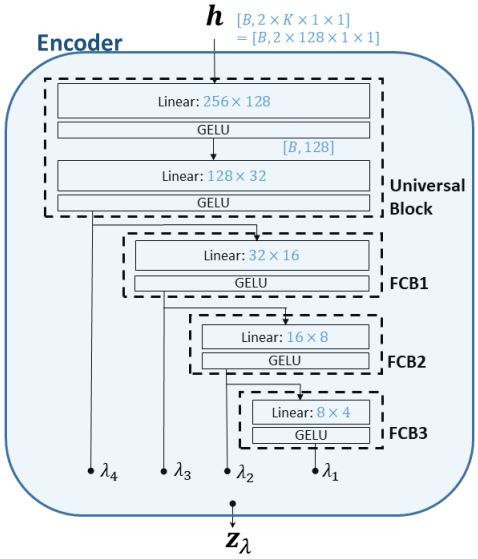
The first baseline is a naive approach which trains the pair of encoder and decoder for each latent vector size $\lambda_i \in \Lambda$ by minimizing the loss function in (5). Fig. 6a shows an example of the MLP architecture using the naive approach to support $\Lambda = \{\lambda_1, \lambda_2, \lambda_3, \lambda_4\} = \{4, 8, 16, 32\}$. This is the worst scheme in terms of the number of parameters.

The second baseline is introduced in [2] which proposes a multiple compression ratio network, named SALDR. It consists of a universal block and serial compression blocks, named fully connected block (FCB), as shown in Fig. 6b. The universal block first extracts CSI features of size $\lambda_{max} = \lambda_4$. Then, FCBs sequentially compress the latent vectors with smaller compression ratios. Employing the universal block can significantly save the HW complexity compared to the naive approach, but it has two limitations. First, as the fully connected layer is sequentially applied, latency to generate the latent vector of the smallest size can be very large during the inference phase. Second, this architecture is difficult to be extended to the case that the cardinality of Λ is large, i.e., the encoder is required to support many cases of compression ratios. This is because one additional element in Λ requires an additional FCB.

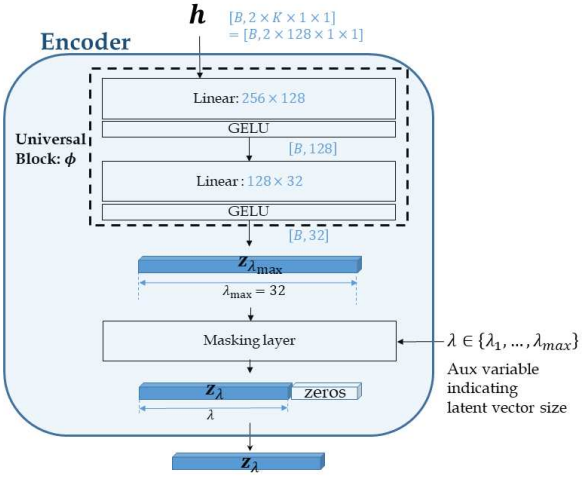
To address these challenges, we propose a new architecture as shown in Fig. 6c. It consists of two parts, universal encoding block and a masking layer. The universal encoding block first extracts CSI features of size λ_{max} which denotes the largest latent vector size that the AE should support. Then, the masking layer bypasses the first λ elements of the output of the universal block and makes the other elements zeros, i.e., auxiliary integer variable λ determines the size of the encoder output. The key idea is that parameters of the universal block are trained such that the element in the earlier position of the latent vector $\mathbf{z}_{\lambda_{max}}$ contains more important information than



(a) Naive Approach



(b) SALDR [2]



(c) Our proposed approach

Fig. 6: Example of block diagram of three approaches to support multiple compression ratios with simple MLP architecture. Universal block in (b) and (c) can be any ML architecture such as CNN or transformer.

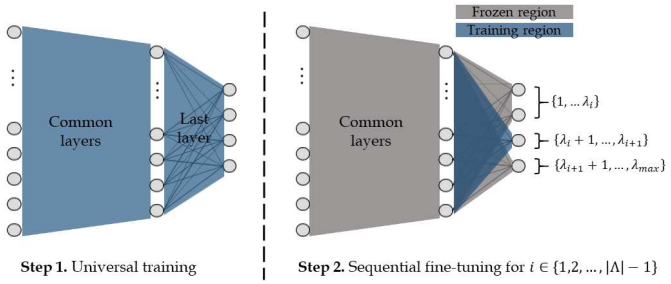


Fig. 7: Two-step training strategy for fine-tuning.

elements in the later position. This enables our architecture naturally select the latent elements from the earlier positions according to the required latent size λ. Our architecture has smaller HW complexity and shorter inference latency as we do not require additional layers for each $\lambda \in \Lambda$. For doing so, we first define the loss function as

$$D_{\phi,\theta_\lambda}(\lambda) = \mathbb{E}_{\mathbf{h}}\left[\left\|\mathbf{h} - g_{\theta_\lambda}(f_\phi(\mathbf{h}) \odot \mathbf{e}_\lambda)\right\|_2^2\right], \tag{7}$$

where $\mathbf{e}_\lambda \in \{0,1\}^{\lambda_{max}}$ is a binary vector whose first λ elements are 1 while the others are 0, and $\odot$ denotes element-wise product. Then the total loss function is defined as

$$D_{\phi,\{\theta_\lambda\}_{\lambda\in\Lambda}}(\Lambda) = \sum_{\lambda\in\Lambda} w_\lambda D_{\phi,\theta_\lambda}(\lambda), \tag{8}$$

where Λ is a set of latent vector size that we need to support. $\{w_\lambda\}_{\lambda\in\Lambda}$ are weight coefficients to present the importance of loss function associated with λ and $\sum_{\lambda\in\Lambda} w_\lambda = 1$. The parameters of encoder and decoder are trained to minimize the total loss function $D_{\phi,\theta}(\Lambda)$, which can be expressed by

$$\phi^{\texttt{opt}}_{\texttt{univ}}, \{\theta^{\texttt{opt}}_\lambda\}_{\lambda\in\Lambda} = \underset{\phi,\{\theta_\lambda\}_{\lambda\in\Lambda}}{\arg\max} \sum_{\lambda\in\Lambda} w_\lambda D_{\phi,\theta_\lambda}(\lambda). \tag{9}$$

Note that $\{w_\lambda\}_{\lambda\in\Lambda}$ plays an important role for the distribution of reconstruction loss over $\lambda \in \Lambda$, and we find the values via hyper-parameter tuning. Reinforcement learning can be utilized to optimize it, which would be one of the interesting future directions.

We further propose a two-step training strategy to improve the reconstruction performance of each compression ratio by leveraging the idea of freezing and fine-tuning in transfer learning [6], [7]. The universal block can be divided into two parts, common layers and the last layer, as depicted in Fig. 7. Key intuition behind the two-step training strategy is that parameters of the common layers contribute to the all compression ratios while parameters of the last layer (commonly fully-connected layer) contribute to a certain compression ratio. In the first step, entire parameters in the common and last layers are trained with the objective function of (9). In the second step, the parameters of the common layers are frozen and partial parameters of the last layer are sequentially trained with $|\Lambda|$ sub-steps. In i-th sub-step ($i \in \{1,\ldots,|\Lambda|\}$), the parameters of the last layer linked to elements of the latent vector with element indexes in $\{\lambda_{i-1}+1,\ldots,\lambda_i\}$ are trained

TABLE II: Summary of settings to generate dataset.

	Settings	# settings.
channel profile	EPA, EVA, TDL [9]	8
SNR (dB)	$11, 12, \ldots, 40$	40
K	$68, 72, \ldots, 128$	16
BS antenna index	$1, 2, \ldots, 32$	32
UE antenna index	$1, 2, 3, 4$	4

to minimize the reconstruction loss of $D_{\phi,\theta_{\lambda_i}}(\lambda_i)$ in (8) while the other parameters are frozen. We observe that this two-step training strategy is especially effective to complicated ML architectures such as transformer-based network [8], which will be detailed in Section IV.

IV. Simulation Results

In this section, we empirically demonstrate that our proposed AE framework has comparable performance in terms of distortion-compression ratio trade-off while it significantly reduces the HW and time complexity in the UE.

Dataset and pre-processing. For a training and test dataset, we generate channel tensors according to the 3rd Generation Partnership Project (3GPP) standard [9] with various resource block size (up to $K = 128$), the number of BS and UE antennas (up to $N_{BS} = 32$, $N_{UE} = 4$), signal to noise ratio (SNR), and channel profile. As described in Section III-A, the generated CSI tensor is partitioned such that each part has a single element in the antenna dimensions, and then each part is zero-padded and transformed to delay domain. The detailed settings are summarized in Table 2, and we randomly generated 12 samples per setting, which results in 5,898,240 samples in total.

Implementations. For the performance evaluation, we implement three approaches that support four latent sizes $\lambda_1 = 4$, $\lambda_2 = 8$, $\lambda_3 = 16$, and $\lambda_4 = \lambda_{\texttt{max}} = 32$, which correspond to the compression ratio of $1/64, 1/32, 1/16, 1/8$ with $K = 128$, respectively. Three approaches are implemented with MLP and the detailed description is included in Section III-B.

- **Approach 1 (Naive approach).** For each λ_i, UE employ and train the encoder ML model ϕ_i separately. This approach is the worst in terms of HW complexity.
- **Approach 2 (SALDR)** [2]. This is the state-of-the-art scheme to support multiple compression ratios, which requires additional fully connected layers to reduce the size of latent vector.
- **Approach 3 (Proposed)**. As the masking layer has no trainable parameter, our proposed approach is the best in terms of the number of parameters (HW complexity) as well as latency (inference time).

Performance comparison. Table III compares the number of parameters and latency of the three approaches to implement MLP encoder(s) to support multiple latent vector sizes $\Lambda = \{4, 8, 16, 32\}$ and $\Lambda = \{1, 2, 3, \ldots, 32\}$. For the latency, we measure the inference time to compress a single CSI tensor $\mathbf{H} \in \mathbb{R}^{2\times K(=128)\times N_{BS}(=32)\times N_{UE}(=4)}$ in the worst scenario. For each compression ratio, we measure time to carry out forward propagation of the encoder 128 times as $\mathbf{H}$ is partitioned into 128 parts $\mathbf{h}^{\texttt{part}} \in \mathbb{R}^{2\times K\times 1\times 1}$, and then report the longest inference time of the CR among Λ. We run $10,000$ times and report the average of running time.

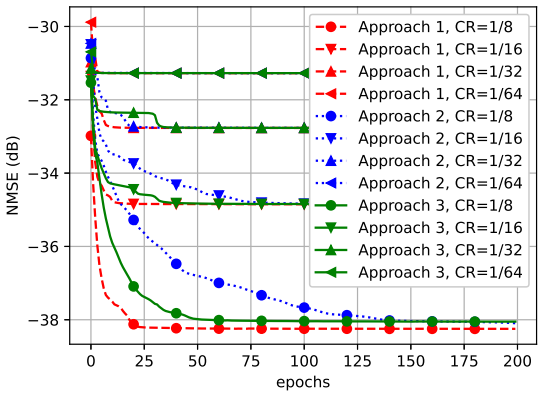


Fig. 8: NMSE performance of three approaches with MLP.

TABLE III: Comparison of HW complexity and latency of three approaches implemented with MLP in two cases with different cardinality of the set of compression ratios, $|\Lambda|$.

	# Params.	Latency (ms)
Approach 1 (Naive)	79.84k	0.1470
Approach 2 (SALDR [2])	37.72k	0.3963
Approach 3 (Ours)	37.02k	0.1455

(a) Case 1: $|\Lambda| = |\{4, 8, 16, 32\}| = 4$,

	# Params.	Latency (ms)
Approach 1 (Naive)	589.1k	0.1446
Approach 2 (SALDR [2])	48.43k	2.708
Approach 3 (Ours)	37.02k	0.1467

(b) Case 2: $|\Lambda| = |\{1, 2, 3, \ldots, 32\}| = 32$.

Fig. 8 shows the NMSE ($\|\mathbf{h} - \hat{\mathbf{h}}\|_2^2/\|\mathbf{h}\|_2^2$) performance of the three approaches with four different compression ratios (CRs). Fig. 9 and Fig. 10 shows the latency and the number of parameters of the three approaches with increasing the cardinality of the set Λ, respectively. We have the following observations:

- Three approaches have the almost same NMSE performance while the proposed approach 3 has much smaller storage and computational complexity than the Approach 1 and 2.
- Our proposed approach reduces the number parameters comparing to Approach 1 (up to $15.9\times$) and Approach 2 (up to $1.3\times$). This gain becomes larger when the cardinality of the set $|\Lambda|$ becomes larger as shown in Fig. 10.
- Our proposed approach significantly reduces the inference time (latency) comparing to Approach 2 because Approach 2 sequentially applies the fully connected layers to reduce the latent vector size. This latency gap becomes larger when the cardinality of the set $|\Lambda|$ becomes larger as shown in Fig. 9.

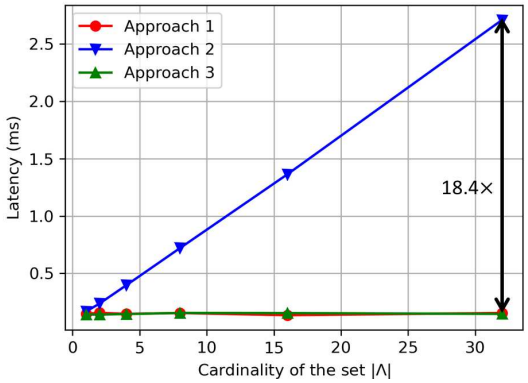


Fig. 9: Inference time (latency) of three approaches with the increasing of the cardinality of the set $|\Lambda|$.

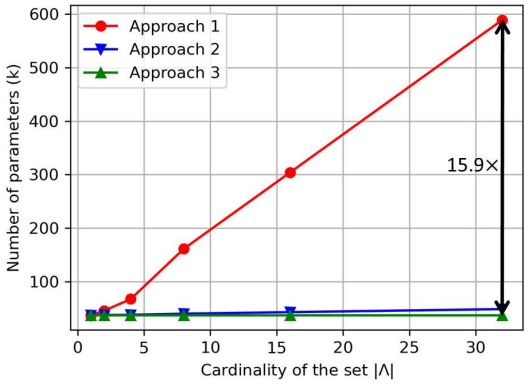


Fig. 10: The number of parameters of three approaches with the increasing of the cardinality of the set $|\Lambda|$.

- One of key contributions of the proposed approach is that HW storage complexity and latency is independent on the cardinality of the set $|\Lambda|$.

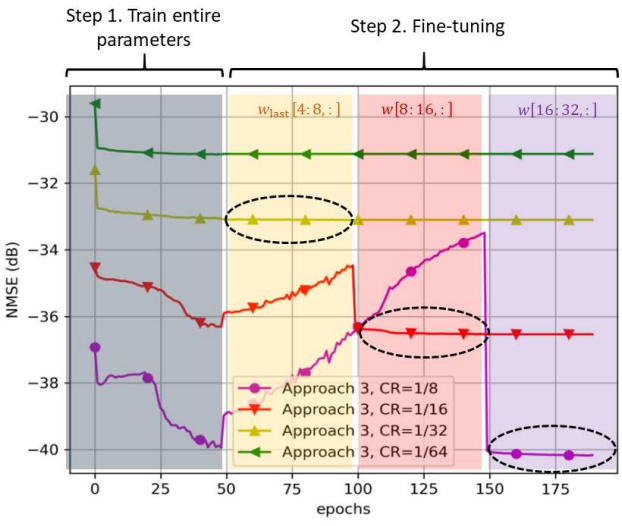


Fig. 11: Learning curve of our proposed approach with two-step training strategy. Transformer-based architecture in [8] is implemented for the universal block depicted in Fig. 6c.

Two-step training strategy. Universal block in the proposed approach can be any ML architecture. When the complicated ML architecture such as a transformer-based network [8] is implemented for the universal block, we observe that learning curve with lower compression ratio (e.g., CR=$1/64$) fluctuates more than the higher CR (e.g., CR=$1/8$). Two-step training strategy described in Section III-B can address this problem. Fig. 11 shows the learning curve of the proposed approach with the two-step training strategy. We can observe that in sub-step of the fine-tuning step, learning curve of the target CR is properly trained while the learning curve of the higher CR remains constant. For instance, in the second sub-step where the target CR is $1/16$ (epochs $\in [100, 150)$), we train the parameters in the last layer of the universal block connected to latent element indexes in $[8, 16)$. Parameters connected to the latent element indexes in $[0, 8)$ is frozen as they have impact on the performance of lower CRs, $1/32$ and $1/64$. Block dotted circles in Fig. 11 indicate the learning curve of the target CR at each sub-step of the fine-tuning step.

V. Conclusion

In this paper, we consider AE-based CSI compression framework with continuous representation in the latent vector. In practice, however, the feedback link has the constraint capability of conveying only finite bits. Investigating how to apply quantization techniques in [10], [11] to our framework would be an interesting future direction.

References

[1] J. Guo, C.-K. Wen, S. Jin, and G. Y. Li, "Overview of deep learning-based CSI feedback in massive MIMO systems," *IEEE Transactions on Communications*, vol. 70, no. 12, pp. 8017–8045, 2022.

[2] X. Song, J. Wang, J. Wang, G. Gui, T. Ohtsuki, H. Gacanin, and H. Sari, "SALDR: Joint self-attention learning and dense refine for massive MIMO CSI feedback with multiple compression ratio," *IEEE Wireless Communications Letters*, vol. 10, no. 9, pp. 1899–1903, 2021.

[3] Y. Xu, M. Yuan, and M.-O. Pun, "Transformer empowered CSI feedback for massive MIMO systems," in *2021 30th Wireless and Optical Communications Conference (WOCC)*. IEEE, 2021, pp. 157–161.

[4] Y. Kim, J. Lee, J. Kim, H. Joo, H. W. Je, and J. Lee, "Channel state feedback with neural networks: A discrete representation learning approach," in *2022 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2022, pp. 184–189.

[5] H. Kim, H. Kim, and G. De Veciana, "Learning variable-rate codes for CSI feedback," in *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE, 2022, pp. 1435–1441.

[6] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, no. 1, pp. 1–40, 2016.

[7] Y. Guo, H. Shi, A. Kumar, K. Grauman, T. Rosing, and R. Feris, "Spottune: transfer learning through adaptive fine-tuning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4805–4814.

[8] S. Mourya, S. Amuru, and K. K. Kuchi, "A spatially separable attention mechanism for massive MIMO CSI feedback," *IEEE Wireless Communications Letters*, 2022.

[9] 3GPP, "Technical specification group radio access network; Study on channel model for frequencies from 0.5 to 100 GHz," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.901, 2017.

[10] S. Ravula and S. Jain, "Deep autoencoder-based massive MIMO CSI feedback with quantization and entropy coding," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 1–6.

[11] K. Kong, W.-J. Song, and M. Min, "Knowledge distillation-aided end-to-end learning for linear precoding in multiuser MIMO downlink systems with finite-rate feedback," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 10, pp. 11 095–11 100, 2021.