IEEE ComSoc | IEEE COMPUTER SOCIETY | IEEE Signal Processing Society | VTS | IEEE Transactions on Machine Learning in Communications and Networking

Received 15 April 2025; revised 10 September 2025 and 16 November 2025; accepted 20 November 2025.
Date of publication 28 November 2025; date of current version 31 December 2025.

The associate editor coordinating the review of this article and approving it for publication was G. Yu.

Digital Object Identifier 10.1109/TMLCN.2025.3638983

Clustered Federated Learning to Support Context-Dependent CSI Decoding

HEASUNG KIM (Member, IEEE), HYEJI KIM (Senior Member, IEEE), AND GUSTAVO DE VECIANA (Fellow, IEEE)

Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX 78712 USA

CORRESPONDING AUTHOR: H. KIM (heasung.kim@utexas.edu)

This work was supported in part by the National Science Foundation under Grant 2148224 and Grant 2443857; in part by the Army Research Office (ARO) under Award W911NF2310062; in part by the Office of Naval Research (ONR) under Award N000142412542; in part by Office of the Under Secretary of Defense for Research and Engineering (OUSD R&E), National Institute of Standards and Technology (NIST), and industry partners as specified in the Resilient and Intelligent NextG Systems (RINGS) Program; and in part by Wireless Networking and Communications Group (WNCG)/6G@UT.

ABSTRACT Neural network-based encoders and decoders have demonstrated significant performance gains over traditional methods for Channel State Information (CSI) feedback in MIMO communications. However, key challenges in deploying these models in real-world scenarios remain underexplored, including: a) the need to efficiently accommodate diverse channel conditions across varying contexts, e.g., environments, and whether to use multiple encoders and decoders; b) the cost of gathering sufficient data to train neural network models across various contexts; and c) the need to protect sensitive data regarding competing providers' coverages. To address the first challenge, we propose a novel system using context-dependent decoders and a universal encoder. We limit the number of decoders by clustering similar contexts and allowing those within a cluster to share the same decoder. To address the second and third challenges, we introduce a clustered federated learning-based approach that jointly clusters contexts and learns the desired encoder and context cluster-dependent decoders, leveraging distributed data. The clustering is performed efficiently based on the similarity of time-averaged gradients across contexts. To evaluate our approach, a new dataset reflecting the heterogeneous nature of the wireless systems was curated and made publicly available. Extensive experimental results demonstrate that our proposed CSI compression framework is highly effective and able to efficiently determine a correct context clustering and associated encoder and decoders.

INDEX TERMS Channel state information, CSI, clustering, compression, federated learning, feedback.

I. INTRODUCTION

THE surging demand for high data rates and the need for optimal interference management in 5G and beyond networks has highlighted the pivotal role that downlink Channel State Information (CSI) feedback plays in enabling Base Stations (BSs) to make informed decisions about signal design. Traditional CSI feedback methods based on codebooks cannot fully utilize the subcarrier correlation and often fall short in MIMO and Massive MIMO communications typically operating over hundreds of subcarriers in 4G LTE [2] or even thousands in 5G NR [3], resulting in a CSI feedback containing more than thousands of bits leading to significant feedback overhead.

To address these limitations, recent research has proposed data-driven CSI feedback methods [4], [5], [6] based on autoencoders, a type of neural network that involves an encoder and decoder [7]. These approaches require collecting CSI data specific to a given *context*, e.g., rural or urban BSs, types of devices, mobility or stability, geographic region, and more. The autoencoder is then trained on a context-specific dataset to compress CSI into a limited-length code using the encoder component deployed in the UE. Subsequently, the decoder component deployed in the BS reconstructs the original CSI from the code. These data-driven context-specific methods have shown substantial performance enhancements compared to traditional approaches like codebook searching and compressive sensing, attracting significant interest in 3GPP Release 18 [8].

While data-driven CSI feedback methods offer significant advantages, existing system models based on autoencoders often fail to directly tackle the heterogeneous nature of channel distributions across different contexts. This oversight can

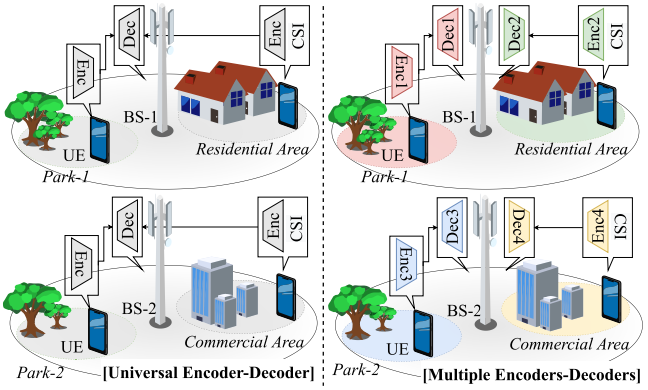

FIGURE 1. The left figure illustrates the use of a global model consisting of a single encoder and decoder for all contexts. On the right-hand side, the deployment of local models is depicted, where a distinct autoencoder is used for each context.

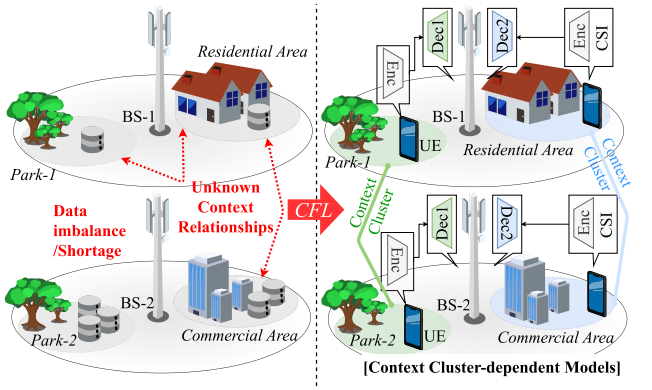

FIGURE 2. The proposed system model groups local datasets into clusters and provides a tailored decoder for each context cluster while learning a universal encoder for any context. This approach overcomes data scarcity and prevents overfitting, resulting in more efficient and accurate solutions. It also eliminates the need for UEs to store multiple encoder models.

potentially lead to performance degradation. Below we start by re-evaluating various possible frameworks for CSI coding.

A. CSI CODING FRAMEWORKS

Universal Encoder-Decoder: Conventional data-driven CSI feedback methods involve devising a single global autoencoder to be used in various contexts. Thus all UE within the system model's coverage is equipped with a single universal encoder, while the BS is also equipped with a universal decoder, regardless of the context, as shown on the left of Figure 1. Adopting a universal model capable of accommodating diverse channel distributions requires a higher representational capacity, which can be achieved through the use of large models and/or an *increased number of feedback bits*. This places significant demands on system resources, including computational, energy consumption, and feedback overheads, as CSI compression and decoding occur frequently within short time intervals, often a few milliseconds. Moreover, this approach may fail to address *dataset imbalances*. For instance, if the dataset associated with a specific context is limited, the performance of the trained autoencoder may suffer for that particular context.

Multiple Encoder-Decoder pairs: To overcome the inefficiencies associated with using a universal encoder-decoder, some existing approaches involve the development of multiple autoencoders, each optimized for a specific context as shown on the right of Figure 1. In this case, the UEs operating in multiple contexts would require multiple encoders, and the BS would in turn require multiple decoders (see Appendix A for a detailed discussion of related work).

In real-world scenarios, managing as many autoencoders as there are contexts is impractical due to the sheer number of different contexts requiring the UEs and BS to store, use, and exchange multiple models. This may be particularly challenging for UEs with limited storage resources. Moreover, collecting sufficient amounts of noise-free CSI data for all contexts may be prohibitively costly, and limited local datasets can lead to overfitted models that do not adequately generalize to real-world tasks.

Context Cluster-Dependent Decoders and a Universal Encoder: To address the limitations of accommodating numerous decoders and encoders for different contexts and considering the potential similarity in CSI distributions among various contexts, we propose a novel framework which groups local datasets (contexts) across the BSs into multiple clusters and provides a tailored decoder for each context cluster with a universal encoder for *any* context, as depicted in Figure 2. The proposed system model offers significant advantages over existing models as follows.

Sample efficiency. By grouping specific contexts and their associated datasets, a robust model with better generalization performance can be provided, overcoming dataset scarcity. This is due to the fact that datasets associated with different contexts can still have similarities in their channel distributions, e.g., BS-1's residential area and BS-2's commercial area in Figure 2.

Codeword usage. In addition, a decoder tailored to a specific cluster of contexts is likely to be more compact and require fewer feedback bits than a universal decoder, offering advantages in terms of reduced computational load and feedback overhead.

Storage efficiency. In this framework, the UEs use a universal encoder for all contexts, allowing them to install only a single neural encoder for storage efficiency and avoiding the need to switch encoders for each context.

B. CONTRIBUTIONS AND ORGANIZATION OF PAPER

System model and algorithm. Our primary contribution lies in proposing the use of a context cluster-dependent CSI compression framework and algorithm to implement the proposed approach. To achieve this, we need to address fundamental questions concerning ***how to cluster contexts*** and ***how to train models in a distributed setting***. It is important to note

that the direct exchange of datasets among BSs or a Central Unit (CU) is often prohibited due to operators' sensitivity to disclosing competitive data or the large size of the CSI instance. In addition to the limited accessibility of the entire raw local datasets, establishing a consistent definition of contexts across BSs or providers poses a challenge when attempting to cluster the contexts.

We formulate the problem of learning context cluster-dependent decoders with a shared encoder as a *Clustered Federated Learning* (CFL) problem and use a novel algorithm CFL-GP (Clustered Federated Learning via Gradient-based Partitioning), which leverages the gradient information generated by local datasets to identify the cluster identities of the contexts. CFL-GP periodically collects gradient information, aggregates it, and groups contexts together based on similarities in their gradients across multiple steps using spectral clustering.

Evaluation. We evaluate our proposed framework and algorithm in diverse settings using COST2100 and 3GPP-3D channel model-based datasets. Our experiments demonstrate the effectiveness of our approach in achieving context clustering without relying on prior knowledge of the contexts. By leveraging this context clustering capability, the proposed method demonstrates substantial enhancements in compression performance, outperforming existing system models in terms of normalized mean squared error for distortion, and significantly improves sample efficiency.

Open-sourced algorithm and dataset. We open-source both the algorithm implementation and the complete dataset [9], ensuring reproducibility and facilitating future research in this area.

The rest of this paper is organized as follows. In Section II, we provide preliminaries on training data-driven CSI feedback encoder/decoder and introduce our autoencoder architecture. Section III presents our system model and formulates the corresponding CFL problem. Our proposed method is described in Section IV. The numerical evaluation of the proposed method is presented in Section V, and the paper concludes in Section VII. Related work can be found in Appendix A.

II. AUTOENCODER-BASED CSI FEEDBACK

For a given BS and a UE pair, we consider a MIMO channel over N_{sc} subcarriers. The BS and UE each have M and a single antenna, respectively. For each subcarrier, we consider an AWGN channel and the output signal $y_n \in \mathbb{C}$ for the input signal $\mathbf{v}_n \in \mathbb{C}^M$ is given as

$$y_n = \mathbf{h}_n^{\mathrm{H}} \mathbf{v}_n + z_n, \qquad 1 \le n \le N_{\mathrm{sc}}, \tag{1}$$

where $\mathbf{h}_n \in \mathbb{C}^{M\times 1}$ is a channel vector, $^{\mathrm{H}}$ indicates Hermitian transpose, and $z_n \in \mathbb{C}$ is the corresponding AWGN. As there are M transmitting antennas and N_{sc} subcarriers involved in the communication, we can gather the channel vectors to form a matrix in the spatial-frequency domain. This matrix is denoted $\mathbf{H}_{\mathrm{SF}} = (\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{N_{\mathrm{sc}}})$ hence $\mathbf{H}_{\mathrm{SF}} \in \mathbb{C}^{M\times N_{\mathrm{sc}}}$.

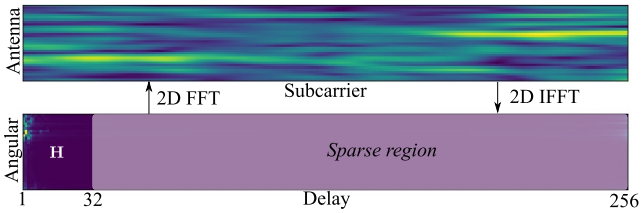

FIGURE 3. ***CSI preprocessing.*** **The matrix in the first row represents the absolute values of a randomly selected CSI matrix $\mathbf{H}_{\mathrm{SF}} \in \mathbb{C}^{32\times 256}$. By applying the 2D-IFFT, we obtain the corresponding angular-delay domain information $\mathbf{H}_{\mathrm{AD}}$. To reduce the computational load, the sparse region is cropped, and the remaining part is normalized and utilized as the ground-truth CSI, denoted as $\mathbf{H} \in \mathbb{C}^{32\times 32}$ with $N_{\mathrm{sc}} = 256$, $M = 32$, and $N_{\mathrm{nz}} = 224$.**

A. CSI PREPROCESSING

Explicit CSI feedback is performed by compressing the channel matrix $\mathbf{H}_{\mathrm{SF}}$ and sending it to the BS. In modern communication systems, typically hundreds or even thousands of subcarriers are used for a communication link, resulting in $\mathbf{H}_{\mathrm{SF}}$ being composed of thousands of elements. Considering this, dealing with such potentially large matrices using a CSI compressor is computationally inefficient. To overcome this problem, we employ the following preprocessing method that exploits the sparsity of CSI in the angular-delay (AD) domain [4]: the information in the spatial-frequency domain is initially transformed through the 2D Inverse Fast Fourier Transform (IFFT) into sparse information in the angular-delay domain, $\mathbf{H}_{\mathrm{AD}}$. In MIMO communications, $\mathbf{H}_{\mathrm{AD}}$ usually involves a sparse region, as shown in prior research [4], [5], [10], where matrix elements tend to be close to zero. To reduce the computational load on the UE, we retain the initial $N_{\mathrm{sc}} - N_{\mathrm{nz}}$ vectors from $\mathbf{H}_{\mathrm{AD}}$, with N_{nz} representing the size of the sparse region which is eliminated. The resulting segment is normalized and referred to as $\mathbf{H} \in \mathbb{C}^{M\times(N_{\mathrm{sc}}-N_{\mathrm{nz}})}$, as illustrated in Figure 3. Each UE is able to estimate the channel matrix $\mathbf{H}_{\mathrm{SF}}$ through pilot signals transmitted from the BS to the UE. We assume that UEs have a perfect CSI $\mathbf{H}_{\mathrm{SF}}$ without an estimation error and focus on the compression problem.

We regard $\mathbf{H}$ as the ground-truth CSI representation, which is subsequently compressed by the UE's encoder and fed back to the BS. The BS receives this feedback information through the wireless link and employs the decoder to reconstruct the original CSI. The process involves adding a zero matrix to the reconstructed data to represent the cropped sparse region and applying the 2D-FFT to obtain the spatial-frequency domain information.

B. AUTOENCODER

An autoencoder is an artificial neural network architecture consisting of two main components: an encoder and a decoder. The encoder part is responsible for compressing a given input instance into a smaller-sized codeword, while the decoder part aims to reconstruct the original instance from

the codeword. The encoder is deployed in the UE, while the decoder is deployed in the BS. This setup allows the UE to compress the CSI matrix $\mathbf{H}$ into a limited-size codeword $\mathbf{C}$ of size N_{cl}, represented in either real-valued or binary form. Subsequently, the BS can decode the codeword $\mathbf{C}$ to estimate the CSI matrix $\hat{\mathbf{H}}$. This information can be utilized for various purposes, such as precoder design or interference mitigation.

We denote the encoder's trainable parameters as $\boldsymbol{\theta}_{\text{enc}}$ and the corresponding function as $g_{\text{enc}}(\mathbf{H}; \boldsymbol{\theta}_{\text{enc}})$. Similarly, the decoder, parameterized by $\boldsymbol{\theta}_{\text{dec}}$, takes the codeword $\mathbf{C} = g_{\text{enc}}(\mathbf{H}; \boldsymbol{\theta}_{\text{enc}})$ and outputs the reconstructed CSI, $\hat{\mathbf{H}} = g_{\text{dec}}(\mathbf{C}; \boldsymbol{\theta}_{\text{dec}})$. We propose an autoencoder architecture inspired by the Vector Quantised-Variational AutoEncoder (VQ-VAE) framework [11] and convolutional networks with residual blocks [12]. A detailed description of the architectural design can be found in Appendix B.

C. AUTOENCODER TRAINING

The autoencoder is trained using gradient-based end-to-end learning. The set of trainable parameters, $\boldsymbol{\theta} = (\boldsymbol{\theta}_{\text{enc}}, \boldsymbol{\theta}_{\text{dec}})$, where $\boldsymbol{\theta} \in \mathbb{R}^d$ and d is the total number of parameters, is updated in the direction of minimizing the Mean Squared Error (MSE) loss

$$L(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{H}}[\ell(\mathbf{H}, \hat{\mathbf{H}})] \tag{2}$$

where ℓ is the squared loss function with respect to the ground-truth CSI and the estimated CSI as follows.

$$\ell(\mathbf{H}, \hat{\mathbf{H}}) = ||\mathbf{H} - \hat{\mathbf{H}}||_{\text{F}}^2 \approx ||\mathbf{H}_{\text{SF}} - \hat{\mathbf{H}}_{\text{SF}}||_{\text{F}}^2. \tag{3}$$

$||\cdot||_{\text{F}}$ indicates the matrix Frobenius norm. It can be interpreted as an approximation of the squared loss of channels in the spatial frequency domain or channels in the angular delay domain since the Fourier transform matrix is unitary.

III. SYSTEM MODEL AND PROBLEM FORMULATION

We consider a distributed framework that involves the collaboration of *multiple* BSs in learning context cluster-dependent decoders and a universal encoder. In the subsequent subsections, we introduce the key components of the system.

A. CONTEXT

The term *context* refers to a characteristic specific to a local dataset, and a BS can encompass multiple contexts. For instance, in Figure 4, BS-1 comprises three contexts: c_1, c_2, and c_3 (Commercial Area, Park-1, and Residential Area-1, respectively). Defining contexts based on UE location is one of the natural criteria, where each context corresponds to a distinct local dataset. Contexts can exist based on various criteria, including not only the location of the UE but also the UEs' speed, antenna configuration, and more. The CSI dataset associated with the c-th context is sampled from a distribution denoted as p_c.

In modern communication systems, such contextual information (e.g., UE configuration, cell association, or coarse location) is routinely available at the BS through control signaling. Leveraging this information, the BS can flexibly select the appropriate decoder (after the training phase) associated with the detected context cluster.

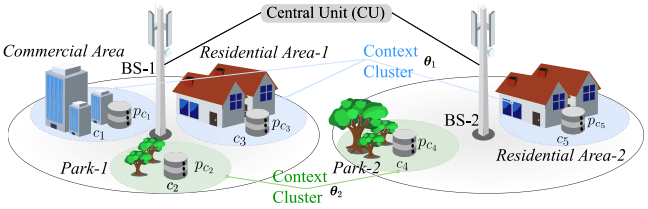

FIGURE 4. An example scenario where 5 different contexts and 2 context clusters exist. Each context cluster shares a decoder. Our goal is to achieve those proper clustering under distributed learning frameworks and develop a universal encoder for all the contexts and the context cluster-dependent decoders.

B. CENTRAL UNIT, BASE STATIONS, AND LEARNING PROTOCOL

The proposed system architecture consists of a CU that communicates with multiple BSs, as exemplified in Figure 4, where the CU connects to BS-1 and BS-2. We employ a Federated Learning (FL) protocol, which is widely recognized as an effective distributed optimization method [13].

We assume that each BS maintains a local dataset of CSI samples for its contexts. These datasets remain local and are not shared. Instead, a CU establishes a wireless or wired connection with the BSs. The CU can transmit models and request gradient computations. Each BS can perform local updates of the model by using the corresponding local dataset.

To maintain practical relevance, we assume that the CU lacks prior knowledge of the channel distributions associated with each context, primarily due to the limited size of the dataset available. Moreover, when privacy-preserving protocols are in place, the CU may be unable to access information about the channel quality or distributions of BSs from competing service providers, further complicating knowledge sharing.

C. TRAINING PIPELINE

Figure 5 illustrates the data and model flow in the proposed framework. The ground-truth channel information $\mathbf{H}$ is required at the BSs for training. This data can be obtained by requesting high-resolution CSI from UEs through dedicated feedback, for example via pilot training where $\mathbf{H}_{\text{SF}}$ is estimated at the UE from downlink pilot signals transmitted by the BS [14]. Alternatively, synthetic channels generated using standardized/advanced channel models can be employed to provide the training data [15], [16].

In the system model, the CU and BSs are the principal training entities, rather than the UEs. As shown in Figure 5, the message flow highlights two key aspects: (i) the transfer of model parameters from the CU to BSs, and (ii) the transfer of the model gap, i.e., the discrepancy between locally trained models, from the BSs to CU.

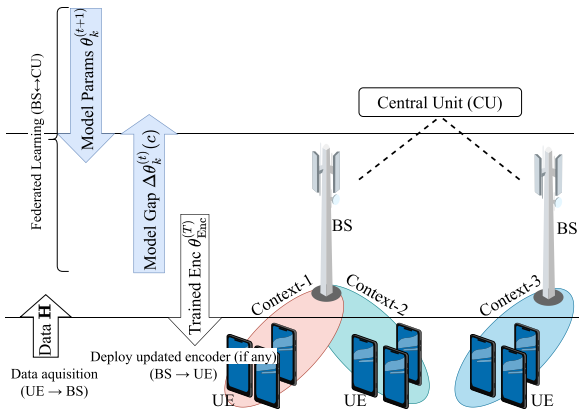

FIGURE 5. Illustration of the message flow in the learning pipeline, including data acquisition, federated training, model parameter transmission from the CU to BSs, and model gap (update information) feedback from the BSs to the CU.

In this work, we consider multiple operational scenarios. In some cases, only the context-dependent decoder at the BS is updated (Sec. V-C), whereas in others, the UE's encoder also requires training (Sec. V-B). In the latter case, the BS can train the encoder model and subsequently deliver it to the UE over the air, corresponding to signaling-based collaboration with model transfer [17], as depicted in Figure 5. An alternative scenario involves the UE directly participating in the model update; however, this approach imposes substantial computational and energy overhead on the UE, though the question of which entity should be responsible for training and updating the model remains open and is the subject of ongoing discussions within both standardization bodies and the research community [18].

D. CSI FEEDBACK AUTOENCODER FRAMEWORKS: CONTEXT CLUSTER-DEPENDENT DECODERS AND A UNIVERSAL ENCODER

In this system model, we reduce the number of decoders by allowing contexts with similar characteristics to share decoders, while employing a universal encoder that is shared across all contexts. To achieve this, we utilize a CFL algorithm, which groups contexts into clusters. Contexts within the same cluster collaboratively train a decoder customized for that cluster. To operate under practical constraints, the CFL algorithm must be capable of clustering the contexts into groups using the available information within the FL framework, without requiring access to raw datasets.

More formally, when the CFL algorithm generates K context clusters, it results in K distinct autoencoders, each associated with one or more contexts. The parameters for the k-th autoencoder are denoted as $\boldsymbol{\theta}_k = (\boldsymbol{\theta}_{\text{enc}}, \boldsymbol{\theta}_{k,\text{dec}})$, where $\boldsymbol{\theta}_{\text{enc}}$ represents the parameters of the universal encoder, and $\boldsymbol{\theta}_{k,\text{dec}}$ represents the parameters of the k-th decoder. Figure 4 illustrates an example scenario involving two models ($K = 2$): $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$. For example, when a UE encounters channels associated with contexts c_1, c_3, or c_5, which correspond to commercial and residential areas, the codeword generated by the universal encoder is decoded using the decoder associated with $\boldsymbol{\theta}_1$.

E. PROBLEM FORMULATION

Our ultimate goal is to cluster a total of C contexts into K context clusters, where $K \leq C$, and train distinct decoders for each cluster while also providing a universal encoder. This approach allows us to enhance performance by offering customized models to each context cluster instead of using a single global model for all contexts. Furthermore, contexts with limited datasets can benefit from being grouped with other contexts for learning, overcoming data scarcity and resulting in better generalization.

For a given context c and a model $\boldsymbol{\theta}_k$, we define the population loss function $F(p_c, \boldsymbol{\theta}_k)$ that represents the expected loss over a corresponding channel distribution p_c. Formally, we have

$$F(p_c, \boldsymbol{\theta}_k) = \mathbb{E}_{\mathbf{H}\sim p_c}[\ell(\mathbf{H}, g_{\text{dec}}(g_{\text{enc}}(\mathbf{H}; \boldsymbol{\theta}_{\text{enc}}); \boldsymbol{\theta}_{k,\text{dec}}))]. \quad (4)$$

Now we present the optimization problem for learning context cluster-dependent decoders and a universal encoder as a CFL problem as follows.

Problem 1 (Clustered Federated Learning): *Given a number of clusters K where $K \leq C$, we aim to find a clustering $\pi : [C] \to [K]$ and K models $\{\boldsymbol{\theta}_k\}_{k=1}^K$ which minimizes the sum loss, i.e.,*

$$\underset{\{\boldsymbol{\theta}_k\}_{k=1}^k, \pi\in\Pi}{\text{minimize}} \quad \sum_{c=1}^{C} F(p_c, \boldsymbol{\theta}_{\pi(c)}) \quad \text{(P1a)}$$

where $\Pi = \{\pi : [C] \overset{\text{S}}{\to} [K]\}$ represents a surjective function space. Each context c is assigned to its respective cluster (model), and the model index of the c-th context can be denoted by $\pi(c)$. We aim to solve Problem P1 by utilizing a CFL algorithm within the systematic constraints outlined in Section III, including the unavailability of the raw dataset for sharing and the unknown channel distribution relationships.

IV. CLUSTERED FEDERATED LEARNING VIA GRADIENT PARTITIONING

In this section, we present a CFL algorithm that allows for context clustering without the requirement of prior knowledge regarding data distributions or sharing of the raw datasets. We extend Clustered Federated Learning via Gradient-based Partitioning (CFL-GP) [1] that utilizes the gradient, i.e., model-update, similarity among contexts with respect to a given set of models as a clustering criterion. We periodically assess the similarity and subsequently partition the contexts into K clusters. The proposed approach is motivated by the intuitive expectation that contexts with similar data distributions should exhibit similarity in gradients calculated from a particular model. It enables efficient, privacy-preserving identification of context clusters without access to raw CSI data.

A. ALGORITHM

Algorithm 1 initializes with a base model and an initial context clustering. If the number of clusters K is known a priori, it is provided as an input. Otherwise, the algorithm starts with $K = 1$ and updates the number of clusters during training. In the latter case, the binary flag `unknown_K` is set to `True` and passed to the spectral clustering module. The k-th context cluster at round t, denoted as $\mathcal{S}_k^{(t)}$, comprises contexts assigned the cluster identity k, i.e., $k_c = k$. Each context is equipped with its own feature vector, initialized as a zero vector and later updated through gradient information. The algorithm requires specifications for the clustering period P and the decision rounds for clustering convergence T_c. These specifications determine how often clustering is performed and when clustering is considered to have converged, respectively. The total number of rounds is denoted as T, and the moving average factor β_t influences the smoothness of feature (averaged gradients) updates. The algorithm's output consists of K trained models, each associated with specific clusters.

Cluster-driven model update (Lines 4-7). The CU updates the models by utilizing the contexts belonging to each of the K clusters. The CU transmits the k-th model, denoted as $\boldsymbol{\theta}_k^{(t)}$, to the BSs associated with the contexts that have cluster index k. Upon receiving the model, each BS performs a local update for each of its contexts. Notably, if a BS manages multiple contexts associated with the k-th model at t, the local update is executed separately for each context (as indicated in Line 5 in Algorithm 1). As described in Algorithm 2, the local update involves T_l-step updates using the local dataset associated with the c-th context. The subroutine GradientUpdate represents gradient-based methods such as stochastic gradient descent or Adam [19], applied to minimize the mean squared error loss over the minibatch $\mathcal{X}_c^{(t_l)}$, which contains the set of CSI instances used for the t_l-th local update of context c. This subroutine returns the difference between the initially received model and the locally updated model, denoted as $\Delta\boldsymbol{\theta}(c)$, referred to as the *model gap*.

The k-th model is updated by collecting the model gaps obtained from the contexts, averaging them, and adding the result to the original model. This approach, known as model averaging, is widely used in FL updates [13]. The subroutine EncoderAveraging($\{\boldsymbol{\theta}_k^{(t+1)}\}_{k=1}^K$) in Line 7 simply indicates that the parameters of the universal encoder is updated by averaging the parameters of the encoders of the K models. This process ensures that all the autoencoders share the same encoder part.

Cluster update (Lines 8-14). At every P iteration (Line 8), if $t < T_c$, the CU updates the clustering of the contexts $\{\mathcal{S}_k^{(t)}\}_{k=1}^K$. Context clustering relies on the direction in which each context updates a set of models. To achieve this, a feature vector $\boldsymbol{g}_c^{(t)} \in \mathbb{R}^{Kd}$ (Kd-dimensional vectors) is maintained for each context and updated at regular intervals. If K is not predefined at the initialization phase, the dimension of the feature vector is dynamically resized as K increases. The *similarity in the model update directions across the K models* for each context is measured using this feature, which serves as the clustering criterion.

More specifically, for each clustering period, we select one of the K models as a model to be broadcasted and send it to all the contexts. We denote the index of this broadcast model as $\bar{k}$, and it is cyclically rotated in a round-robin manner with each clustering period. The CU then requests the model gap for this broadcast model from all contexts (Line 9). If the c-th context has model index $k_c = \bar{k}$ at time t, it does not need to perform a local update again. We use the model gap obtained from the c-th context to update the $\bar{k}$-th block of the feature vector for that context. Specifically, the $\bar{k}$-th block of $\boldsymbol{g}_c^{(t)}$ is $(\bar{k}d - d + 1 : \bar{k}d)$-th elements of $\boldsymbol{g}_c^{(t)}$. This feature is smoothly updated as $\boldsymbol{g}_{c,\bar{k}}^{(t)} \leftarrow (1-\beta_t)\boldsymbol{g}_{c,\bar{k}}^{(t-1)} + \beta_t \Delta\boldsymbol{\theta}_{\bar{k}}^{(t)}(c)$, where β_t denotes the moving average factor (Line 10). All the remaining feature elements, except for the $\bar{k}$-th block, will remain the same.

We utilize Spectral clustering [20] to cluster the contexts based on the feature vectors. The subroutine SpectralClustering in Algorithm 3 consists of three main steps:

Dimensionality reduction. Let $\boldsymbol{G}^{(t)} = (\boldsymbol{g}_1^{(t)} \cdots \boldsymbol{g}_C^{(t)}) \in \mathbb{R}^{Kd \times C}$ represent the matrix of gradient features. In Line 3 of Algorithm 3, LSV_S refers to the process of obtaining S Leading Singular Vectors from $\boldsymbol{G}^{(t)}$. The CU computes $\hat{\boldsymbol{U}}^{(t)} \in \mathbb{R}^{Kd \times S}$, where i-th column corresponds to the singular vector of $\boldsymbol{G}^{(t)}$ associated with the i-th largest singular vector. If K is given, we have $S = K$. The value of S can be determined by observing the singular gaps, which refer to the differences between the singular values of $\boldsymbol{G}^{(t)}$. We can identify consecutive indices that exhibit the maximum gap and select the preceding index as the number of leading singular values. In practice, we ensure $S \geq 3$ to preserve sufficient information from the original feature matrix. The feature matrix $\boldsymbol{G}^{(t)}$ is then projected onto this reduced subspace by multiplying it with the transpose of the leading singular vectors, yielding the reduced features $\boldsymbol{X}$.

K-means clustering. Following dimensionality reduction, the CU performs K-means clustering to partition contexts into K groups. The KMeans subroutine solves the following optimization problem.

$$(\hat{\boldsymbol{z}}, \{\bar{\boldsymbol{w}}_k\}_{k=1}^K) = \underset{\boldsymbol{z}, \{\boldsymbol{w}_k\}_{k=1}^K}{\operatorname{argmin}} \sum_{c=1}^{C} ||\boldsymbol{X}_{:,c} - \boldsymbol{w}_{z_c}||^2 \tag{5}$$

If the number of clusters K is not specified at initialization, we determine it using the Silhouette score [21]. Specifically, we evaluate candidate values of K in the range $[2, C]$, perform K-means clustering for each, and select the value that yields the highest Silhouette score [22]. The Silhouette score evaluates clustering quality by averaging the Silhouette coefficient across all samples. For each sample $\boldsymbol{X}_{:,c}$, the coefficient is defined as $(\phi_{\text{nearest}} - \phi_{\text{intra}})/\max(\phi_{\text{intra}}, \phi_{\text{nearest}})$, where ϕ_{intra} is the mean intra-cluster distance, and ϕ_{nearest} is the mean nearest-cluster distance. A higher score

Algorithm 1 Clustered Federated Learning to Support Context-Dependent CSI Decoding

Input: Number of clusters K, `unknown_K`, an initial model $\boldsymbol{\theta}_1^{(0)}$, initial cluster $\mathcal{S}_1^{(1)} = [C]$, clustering period P, gradient features $\boldsymbol{g}_c^{(-1)} \forall c \in [C]$, feature moving average factor $\{\beta_t\}_{t=1}^{T}$, training round T, clustering termination threshold T_c
Output: K trained models $\{\boldsymbol{\theta}_k^{(T)}\}_{k=1}^{K}$ and K updated clusters $\{\mathcal{S}_k^{(T+1)}\}_{k=1}^{K}$
for $t = 1$ ***to*** T **do**
  for $k = 1$ ***to*** K *in parallel* **do** `/* Model Update */`
    CU transmits $\boldsymbol{\theta}_k^{(t)}$ to the contexts in $\mathcal{S}_k^{(t)}$ and receives $\Delta\boldsymbol{\theta}_k^{(t)}(c) \leftarrow$ LocalUpdate$(\boldsymbol{\theta}_k^{(t)}, c)\ \forall c \in \mathcal{S}_k^{(t)}$
    CU updates $\boldsymbol{\theta}_k^{(t+1)} \leftarrow \boldsymbol{\theta}_k^{(t)} + \frac{1}{|\mathcal{S}_k^{(t)}|}\sum_{c\in\mathcal{S}_k^{(t)}} \Delta\boldsymbol{\theta}_k^{(t)}(c)$
  EncoderAveraging$(\{\boldsymbol{\theta}_k^{(t+1)}\}_{k=1}^{K})$
  if $t \bmod P = 1$ *and* $t < T_c$ **then** `/* Cluster Update */`
    CU broadcasts $\boldsymbol{\theta}_{\bar{k}}^{(t)}$ and receives $\Delta\boldsymbol{\theta}_{\bar{k}}^{(t)}(c) \leftarrow$ LocalUpdate$(\boldsymbol{\theta}_{\bar{k}}^{(t)}, c)\ \forall c \in [C]$
    CU updates $\boldsymbol{g}_{c,\bar{k}}^{(t)} \leftarrow (1-\beta_t)\boldsymbol{g}_{c,\bar{k}}^{(t-1)} + \beta_t \Delta\boldsymbol{\theta}_{\bar{k}}^{(t)}(c)\ \forall c \in [C]$ and $\boldsymbol{g}_{c,k'}^{(t)} \leftarrow \boldsymbol{g}_{c,k'}^{(t-1)} \forall c \in [C], k' \in [K]$ s.t. $k' \neq \bar{k}$
    $(\{\mathcal{S}_k^{(t+1)}\}_{k=1}^{K}, K) \leftarrow$ SpectralClustering$(\{\boldsymbol{g}_c^{(t)}\}_{c=1}^{C}, \{\mathcal{S}_k^{(t)}\}_{k=1}^{K}$, `unknown_K`$)$
    $\bar{k} \leftarrow (\bar{k} \mod K)+1$
  else
    $\mathcal{S}_k^{(t+1)} \leftarrow \mathcal{S}_k^{(t)}\ \forall k \in [K]$ and $\boldsymbol{g}_c^{(t)} \leftarrow \boldsymbol{g}_c^{(t-1)}\ \forall c \in [C]$

Algorithm 2 LocalUpdate

Input: Model $\boldsymbol{\theta}^{(0)}$, context c
Output: Model update $\Delta\boldsymbol{\theta}(c) = \boldsymbol{\theta}^{(T_l)} - \boldsymbol{\theta}^{(0)}$
for $t_l = 1$ ***to*** T_l **do**
  $\boldsymbol{\theta}^{(t_l)} \leftarrow$ GradientUpdate$(\boldsymbol{\theta}^{(t_l-1)}, \mathcal{X}_c^{(t_l)})$

Algorithm 3 SpectralClustering

Input: features $\{\boldsymbol{g}_c^{(t)}\}_{c=1}^{C}$, clusters $\{\mathcal{S}_k\}_{k=1}^{K}$, `unknown_K`
Output: $\{\{c|k_c'=1\}, \cdots, \{c|k_c'=K\}\}, K$
$\hat{\boldsymbol{U}}^{(t)} \leftarrow \mathrm{LSV}_S(\boldsymbol{G}^{(t)})$ if `unknown_K` else $\mathrm{LSV}_K(\boldsymbol{G}^{(t)})$
$\boldsymbol{X} \leftarrow \hat{\boldsymbol{U}}^{(t)\top}\boldsymbol{G}^{(t)}$
$\mathcal{K} \leftarrow \{2, \ldots, C\}$ if `unknown_K` else $\{K\}$
$K \leftarrow \underset{K'\in\mathcal{K}}{\operatorname{argmax}}$ Silhouette$\big($KMeans$(\boldsymbol{X}, K')\big)$
$(\hat{\boldsymbol{z}}, \{\bar{\boldsymbol{w}}_k\}_{k=1}^{K}) =$ KMeans$(\boldsymbol{X}, K)$
$\hat{\psi} \leftarrow \underset{\psi\in\Psi}{\operatorname{argmax}} |\{c : \psi(\hat{z}_c) = k_c\}|$
Set $k_c' \leftarrow \hat{\psi}(\hat{z}_c)\ \forall c$

indicates stronger cohesion within clusters and greater separation between them. We adopt the implementation from [22] for efficient selection of K.

Model Assignment. For each group, a unique model is assigned, yielding K labeled clusters $\{\mathcal{S}_k^{(t+1)}\}_{k=1}^{K}$. To ensure consistency, the model assignment is chosen to maximize alignment with the previous clustering, by selecting a permutation function $\hat{\psi}$ from the bijective function space $\Psi = \{\psi : [K] \to [K]\}$.

If the condition for triggering the clustering process is not satisfied at t, the cluster remains the same as the cluster from the previous step.

B. PRETRAINED ENCODER

Algorithm 1 can make use of a pretrained universal encoder. With a given universal encoder, the algorithm operates with fixed encoder weights throughout the training process, thereby restricting gradient updates only to the decoders' weights. Specifically, this adjustment means that Line 4 in Algorithm 2 is applied exclusively to the decoders, effectively rendering Line 7 of Algorithm 1 redundant. This approach is closely related to concepts like partial model personalization and alternative optimization [23], though our focus remains on context clustering and the cluster-wise personalization.

To obtain a universal pretrained encoder, one could set $K = 1$ and execute Algorithm 1, using the resulting single trained encoder, or consider the use of a predefined standardized encoder.

C. COMMUNICATION COMPLEXITY AND PRACTICALITY

The communication efficiency of FL depends on the system configuration, including model size, number of training rounds, and dataset characteristics. Table 1 compares the training cost and data transmission load between centralized learning and the proposed CFL-based context-dependent approach.

The proposed approach performs context clustering based solely on model gaps (i.e., gradient information), and therefore requires no additional signaling beyond what is already present in conventional FL. The only added overhead is a single extra model update and transmission once every P rounds per context, which terminates after T_c clustering rounds. In terms of storage, the method temporarily requires $Kd \times C$ features-K times larger than conventional FL-but this requirement disappears once clustering converges. In practice, clustering converges within about 10 communication rounds (see Sec. V-C), making the additional computation

TABLE 1. Training and transmission cost: centralized vs. CFL.

Aspect	Centralized Learning	Clustered Federated Learning (CFL)
Model training	$T \cdot \sum_{c=1}^{C} T_l$ model updates processed at CU	$T \cdot T_l$ local updates per context
Data transmission load	$N_{\text{sample}} \cdot M \cdot N_{\text{sc}} \cdot 2$ (without preprocessing) $N_{\text{sample}} \cdot M \cdot (N_{\text{sc}} - N_{\text{nz}}) \cdot 2$ (with preprocessing)	Every P rounds: $d \cdot (P+1)$ per context (Both upstream and downstream)
Data privacy	Raw CSI must be shared with CU (possibly across vendors)	Model update directions (gradients) are exchanged

and communication costs negligible relative to the full training process. As a result, the proposed method remains fully compatible with existing FL protocols, while preserving privacy and incurring only minimal overhead.

Under certain scenarios, FL can reduce communication overhead compared to centralized learning. For instance, in centralized training, each BS must transmit its entire local dataset to a CU. For example, in Sec. V, we consider the COST2100 dataset [4] for indoor and outdoor environments with $N_{\text{sample}} = 100,000$ samples and each raw sample originally drawn from $\mathbb{C}^{32\times1024}$ in the spatial-frequency domain. This results in over 6×10^9 real numbers. With FL, if we train a CsiNet [4] model for $N_{\text{cl}} = 64$ having 2.6×10^5 parameters over 1,000 rounds with $C = 8$, the cumulative transmission cost would be approximately less than 5×10^9 real numbers, a reduction in communication load under this setup. It is important to note that in this setup, it is not the UE but the BS that transmits and receives the model parameters, and thus, we consider the BS as the FL client participating in collaborative training.

However, when larger models are used or centralized learning applies the preprocessing to reduce data transmission, such as delay-domain cropped CSI of dimension $\mathbb{C}^{32\times32}$, centralized learning can be more communication-efficient than FL. The tradeoff between FL and centralized learning varies across scenarios, whereas FL becomes increasingly advantageous as dataset sizes grow, which is typical in modern massive MIMO systems with high-dimensional CSI.

Beyond communication efficiency, FL enables privacy-preserving collaborative training across multiple service providers without requiring the exchange of raw CSI data. This is particularly beneficial in wireless systems where CSI data often resides across competing service providers or UE vendors [17]. The value of such privacy-aware training frameworks has been widely acknowledged in recent studies [24], [25]. Furthermore, as noted in [26], 3GPP is expected to continue its study of CSI compression in Release 19, to address the challenges of inter-vendor training collaboration. In this context, the proposed CFL-based framework can support collaborative learning in multi-vendor (context) environments.

V. EVALUATION

In this section, we first demonstrate that CFL achieves performance comparable to centralized learning with known context clusters, without requiring prior knowledge of the underlying distributions, and by leveraging distributed data. We further show that the CFL approach outperforms both the global and local model schemes discussed in Sec. I. Next, we consider a more challenging and practically relevant setting with heterogeneous channel distributions across contexts. We evaluate the CFL algorithm under this setting and observe that clustering contexts with shared similarities results in improved performance compared to baseline schemes. Additionally, we compare the context clustering results with existing CFL algorithms and show that our method consistently yields correct clustering outputs, leading to better compression performance.

A. BASELINES

1) GLOBAL MODEL

The scheme that utilizes a single global model corresponds to setting K to 1 in the proposed algorithm. This is equivalent to the traditional FL approach [13], where a shared encoder and decoder are used for all contexts.

2) LOCAL MODELS

In the scheme where each context utilizes a different local autoencoder model, it can be achieved by setting K to C in the proposed algorithm and excluding the process described in lines 7-14 of Algorithm 1. This configuration aligns with the conventional data-driven CSI feedback framework, where each context has a distinct autoencoder.

3) EXISTING CFL ALGORITHMS

In selected experiments, we employ existing representative CFL algorithms, Iterative Federated Clustering Algorithm (IFCA) [27] and Clustered Federated Learning: Model-Agnostic Distributed Multi-Task Optimization under Privacy Constraints (MADMO) [28], as baselines. While these CFL algorithms could potentially be adapted to support our proposed system model, their outcomes, including clustering results, can be different from ours.

To measure the compression efficiency, we employ the Normalized Mean Squared Error (NMSE) as a standard metric which is given as $\mathbb{E}[\|\mathbf{H} - \hat{\mathbf{H}}\|_{\text{F}}^2 / \|\mathbf{H}\|_{\text{F}}^2]$.

B. SETUP 1: INDOOR AND OUTDOOR CHANNEL DISTRIBUTIONS

1) SIMULATION ENVIRONMENT

In this subsection, we assume two types of channel distributions: indoor and outdoor, which indicates that there are two different distribution identities as $D = 2$. We use the COST2100 Channel Model Dataset [4] that has been widely

TABLE 2. Performance comparison (NMSE).

	Average				Indoor channel				Outdoor channel			
N_{cl} / C	Context-dep	Global	Local	Centralized [4]	Context-dep	Global	Local	Centralized [4]	Context-dep	Global	Local	Centralized [4]
32 / 8	**-3.40**	-2.47	-2.94	-3.45	**-5.30**	-3.60	-4.66	-5.84	**-2.09**	-1.58	-1.72	-1.93
32 / 16	**-3.05**	-2.36	-2.39	-3.45	**-5.00**	-3.35	-4.06	-5.84	**-1.72**	-1.57	-1.19	-1.93
32 / 32	**-2.83**	-2.53	-1.41	-3.45	**-4.11**	-3.70	-2.83	-5.84	**-1.85**	-1.61	-0.35	-1.93
64 / 8	**-4.45**	-4.31	-4.17	-4.19	-6.91	-6.60	**-6.98**	-6.24	**-2.89**	-2.83	-2.49	-2.81
64 / 16	**-4.64**	-4.33	-3.46	-4.19	**-7.16**	-6.46	-6.50	-6.24	**-3.06**	-2.92	-1.69	-2.81
64 / 32	**-4.56**	-4.19	-2.22	-4.19	**-7.05**	-6.36	-4.84	-6.24	**-3.00**	-2.75	-0.61	-2.81
128 / 8	**-6.28**	-5.78	-5.35	-6.10	**-8.60**	-7.61	-7.89	-8.65	**-4.78**	-4.50	-3.77	-4.51
128 / 16	-5.91	**-5.94**	-4.15	-6.10	**-8.30**	-7.97	-7.23	-8.65	-4.38	**-4.57**	-2.37	-4.51
128 / 32	**-5.83**	-5.60	-2.76	-6.10	-7.66	**-7.70**	-6.15	-8.65	**-4.56**	-4.19	-0.89	-4.51

used as a benchmark for DL-based CSI feedback research. The COST2100 dataset comprises 100,000 instances of indoor channel data and 100,000 instances of outdoor channel data. In order to examine scenarios with a reduced amount of data, we select 50,000 instances from each set, leading to a total dataset of 100,000 instances.

We consider a system model consisting of one CU connected to C contexts, with $K = 2$, corresponding to two distinct channel distribution identities. Among the C contexts, half are associated with an indoor channel distribution, while the other half are linked to an outdoor channel distribution. Each context is assigned a local dataset that contains an equal portion of the overall indoor channel data. The experiments are conducted with different configurations, specifically $C \in \{8, 16, 32\}$ and $N_{cl} \in \{32, 64, 128\}$.

2) ALGORITHMIC CONFIGURATION

The learning rate is set to 0.001 with the Adam optimizer [19], and a batch size of 200 is selected. During each local update, each context leverages its entire local dataset to perform a single epoch update. The local dataset is divided into multiple batches of size 200, and these batches are sequentially used to update the model. We employed the encoder and decoder architectures described in [4].

3) RESULTS

In Table 2, an underlined result indicates the superior performance of a distributed learning approach over the centralized learning approach [4]. The best-performing distributed learning approach is highlighted in **bold**. The performance of the centralized learning approach reported in Table 1 is based on the results presented in [4]. The key distinction between the centralized learning approach and the distributed learning approach lies in the fact that the learning agent (CU) in the centralized approach has knowledge of the distribution identity of all the data and has direct access to the raw data. The centralized framework employs two autoencoders, which may offer enhanced compression capability. All FL-based schemes-including the proposed context-dependent decoding, the global model, and the local model-use encoder/decoder modules that are architecturally identical to those in the centralized framework. However, the proposed context-dependent decoding scheme utilizes a single universal encoder, while the global model scheme employs a single encoder and a single decoder, both operating without access to raw datasets.

The proposed approach, referred to as *Context-dep*, demonstrates superior performance compared to other distributed learning methods across most experimental scenarios. The proposed method successfully clusters contexts into two distinct groups: one containing indoor channel-associated contexts and the other comprising outdoor channel-associated contexts. This clustering is achieved without any prior knowledge of the channel distributions for the contexts.

Using the single global model strategy yields performance variations across diverse setups, with differences within approximately 0.3 dB range for varying context numbers. The approach employing local models for each context experiences significant performance degradation as the number of contexts increased and per-context dataset sizes decreased. This can result in performance gaps exceeding 2 dB, possibly due to data limitations. In contrast, the proposed method consistently outperforms these approaches in all experiments, irrespective of codeword length or context count.

Notably, the proposed method even outperforms the centralized learning method in terms of average performance in some cases, despite the latter having access to all data and knowledge of their distribution identities. Additionally, it is important to highlight that our proposed method employs a universal encoder, reducing the encoder storage requirement for each UE to half that of the centralized learning method.

C. SETUP 2: DIVERSE HETEROGENEOUS CONTEXTS OVER MULTIPLE SUBAREAS AND MULTIPLE FREQUENCIES

In this subsection, we explore a scenario where all the contexts have distinct channel distributions, and we investigate the potential benefits of employing CFL to cluster contexts with distinct channel distributions together. This experimental setup can potentially encompass scenarios where multiple service providers operate in geographically distinct regions, utilizing different frequency bands, and the goal is to enhance the performance of the CSI autoencoders for the various contexts by CFL.

TABLE 3. Comparison of NMSE (dB) performance.

	$c_1, \mathbf{H} \sim p_1$			$c_2, \mathbf{H} \sim p_2$			$c_3, \mathbf{H} \sim p_3$			$c_4, \mathbf{H} \sim p_4$		
N_{cl}	Context-dep	Global	Local	Context-dep	Global	Local	Context-dep	Global	Local	Context-dep	Global	Local
64-bit	**-9.04**	-7.92	2.58e-4	**-10.06**	-7.98	1.97e-4	**-4.71**	-4.05	2.04e-4	**-5.12**	-4.55	4.41e-4
128-bit	**-10.94**	-9.96	-1.38	**-12.13**	-9.86	2.47e-4	**-6.84**	-6.23	2.53e-4	**-7.25**	-6.78	3.76e-4
256-bit	**-11.10**	-10.16	-1.07	**-13.17**	-10.09	-1.58	**-8.81**	-8.12	-1.34	**-9.19**	-8.67	-0.89
	$c_5, \mathbf{H} \sim p_5$			$c_6, \mathbf{H} \sim p_6$			$c_7, \mathbf{H} \sim p_7$			$c_8, \mathbf{H} \sim p_8$		
N_{cl}	Context-dep	Global	Local	Context-dep	Global	Local	Context-dep	Global	Local	Context-dep	Global	Local
64-bit	**-6.59**	-5.36	5.30e-4	**-6.62**	-5.38	1.29e-3	**-1.72**	-1.44	3.26e-4	**-2.33**	-1.78	5.58e-4
128-bit	**-7.69**	-6.77	6.36e-4	**-8.18**	-7.18	5.12e-4	**-3.08**	-2.60	3.21e-4	**-4.01**	-3.26	5.62e-4
256-bit	**-9.41**	-7.86	3.95e-4	**-9.73**	-8.43	-1.42	**-4.53**	-3.53	-1.97	**-5.76**	-4.49	-1.34

1) SIMULATION ENVIRONMENT

We generate eight local datasets, each comprising 10,000 training samples and 10,000 test samples, using QuaDRiGa (QUAsi Deterministic RadIo channel GenerAtor) [29], [30]. The datasets differ in carrier frequency (0.8/2.4 GHz), cell type (Urban Micro/Macro), and propagation condition (LOS/NLOS). LOS channels contain 5–20 dominant delay components, whereas NLOS channels exhibit 30–40 components, resulting in richer multipath propagation. Base stations employ the 3GPP-3D antenna model, while user equipment is equipped with omni-directional antennas. Macro-cell and micro-cell base stations are placed at heights of 24 m and 12 m, respectively. UEs are distributed over a 35 m radius disk at a height of 1.5 m, whose center is located 35 m from the base station along the main lobe direction of the BS antenna. This setup ensures variability in spatial geometry and signal paths, reflecting realistic deployment scenarios.

To further illustrate the diversity among these datasets, Figure 6 visualizes six randomly selected samples from each p_c, represented in the angular-delay domain. Each CSI matrix is represented in the cropped angular-delay domain, normalized by its maximum value before taking the absolute magnitude. These visualizations show heterogeneity across contexts in terms of scattering patterns, angular dispersion, and dominant delay components. However, partial similarities are also observed across some contexts, motivating the use of context clustering to promote decoder sharing within the CFL framework.

2) ALGORITHMIC CONFIGURATION

We use the autoencoder architecture detailed in Appendix B, configured with $B = 16$ and an embedding dimension $N_{\text{emb}} = 4$. The training objective uses the modified loss function ℓ' defined in (6), replacing ℓ, for a discretized codeword space. For the CFL baselines and our proposed method, we initialize the training with a pre-trained global encoder. This universal encoder is obtained via Algorithm 1 with $K = 1$ and $T = 1{,}000$, as described in Subsection IV-B, and the encoder parameters remain fixed throughout the CFL training process. To train the context-dependent CSI decoders, we run $T = 1{,}000$ communication rounds, with $T_c = 10$ and single-epoch local updates. We assume the number of clusters is unknown and accordingly set `unknown_K = True`. All remaining hyperparameters follow the experimental setup outlined in Subsection V-B.

3) RESULTS

Across nine experiments spanning three codeword lengths (64, 128, and 256 bits) and three random seeds, the proposed CFL algorithm consistently produces a single clustering outcome: $\{(c_1, c_2), (c_3, c_4), (c_5, c_6), (c_7, c_8)\}$, where each context cluster is denoted by parentheses, e.g., (c_1, c_2). Figure 7 illustrates the resulting architecture, where a single universal encoder is shared across all contexts, while four distinct decoders are assigned to context clusters. For example, when a user is in context c_1 or c_2, corresponding to channel distributions p_1 and p_2, the UE uses the shared universal encoder and the BS, upon receiving the encoded feedback, employs Dec 1, which is shared between both contexts for CSI reconstruction.

Table 3 provides a comparative analysis of the compression performance across different frameworks for the 64, 128, and 256-bit compression scenarios. The results demonstrate that the proposed context cluster-dependent framework determined by CFL-GP consistently surpasses both the global and local model frameworks across all contexts. Notably, in several settings, our approach achieves superior reconstruction accuracy using only half the feedback budget required by the global model. For example, in context c_1 with $\mathbf{H} \sim p_1$, the proposed approach achieves better performance with 64-bit codewords than the global model with 128 bits. Similarly, for p_2, our method attains an NMSE of -10.06 dB with 64 bits, while the global model, despite utilizing twice the feedback budget, reaches -9.86 dB. The local training approach performs significantly worse across all settings, likely due to the absence of context collaboration and the limited sample size per context. As expected, both the proposed method and the global baseline benefit from increased feedback capacity.

a: CODEBOOK PERPLEXITY

To evaluate the codebook usage, we measure the *codebook perplexity* for each context, as summarized in Table 4. The perplexity quantifies the effective number of codewords being utilized and is defined as $\exp(-\sum_{i=1}^{B} q_i \log q_i)$, where

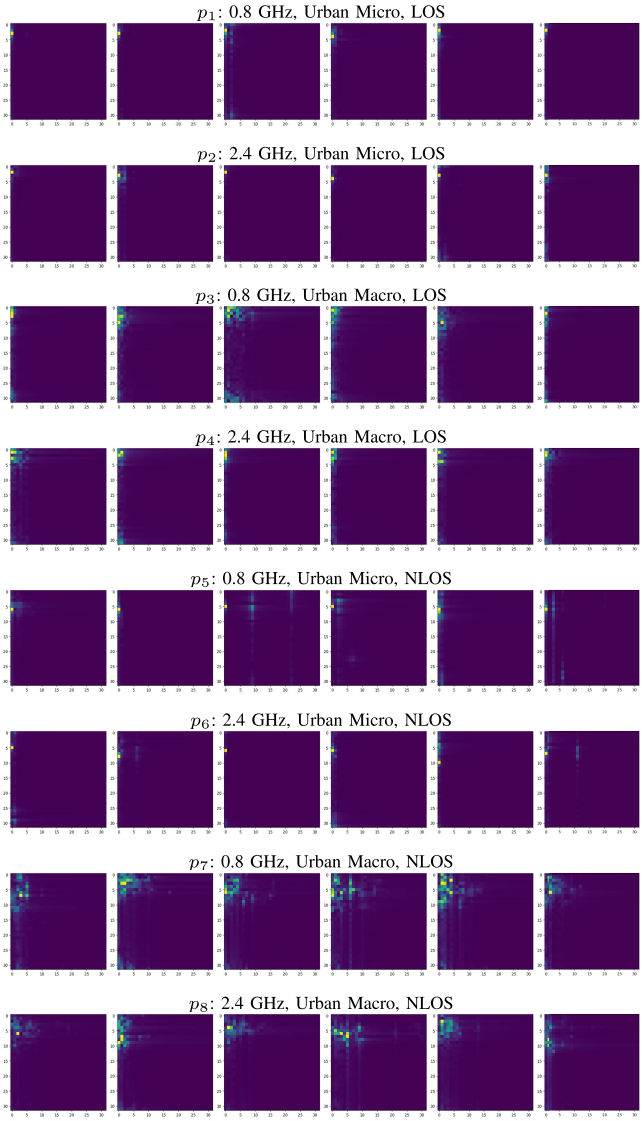

FIGURE 6. Sample visualizations from the heterogeneous channel datasets, $p_1 \ldots p_8$, used in the federated learning experiments. Each dataset differs in center frequency, propagation condition, and cell type. While the datasets exhibit heterogeneity, partial similarities exist across subsets, which motivates context clustering. CFL algorithms aim to identify clusters to assign shared decoders while retaining a universal encoder. The full dataset is publicly available at [9].

TABLE 4. Codebook perplexity.

N_{cl}	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
64-bit	13.54	13.04	15.33	15.32	14.38	14.39	14.00	14.38
128-bit	11.03	10.07	15.22	13.61	13.66	14.36	14.36	14.18
256-bit	11.18	10.48	15.02	14.59	13.39	13.25	14.79	14.77

B denotes the codebook size and q_i represents the empirical probability of selecting the i-th codeword. By construction, the perplexity is between 1 and B, with higher values indicating a more uniform usage of the codebook.

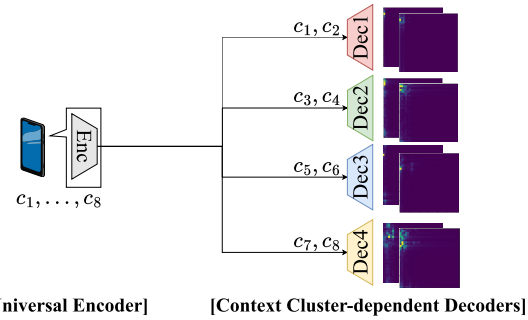

FIGURE 7. Trained context cluster-dependent framework with a single universal encoder and four context cluster-dependent decoders. The proposed CFL algorithm consistently yields the clustering $\{(c_1, c_2), (c_3, c_4), (c_5, c_6), (c_7, c_8)\}$ across all experiments. For instance, contexts c_1 and c_2, corresponding to channel distributions p_1 and p_2, share the same decoder while using the universal encoder.

The results in Table 4 show that most contexts achieve perplexity values in the range of 13–15, which is close to the maximum possible value of 16. This observation indicates that the learned quantizer is effectively leveraging the majority of the available codewords across contexts. In particular, c_3 and c_4 attain perplexity values exceeding 15, reflecting near-uniform codebook utilization. On the other hand, relatively lower values are observed for c_1 and c_2 (e.g., 10.07 at 128 bits), suggesting that their channel distributions are simpler and predominantly mapped to a smaller subset of codewords. This is consistent with their relatively higher compression efficiency, as these contexts correspond to LOS or microcell scenarios with reduced scattering complexity.

4) CONTEXT CLUSTER-DEPENDENT MODELS VIA DIFFERENT CFL ALGORITHMS

Our proposed system model, which leverages context cluster-dependent decoders and a universal encoder, can also be implemented using existing CFL algorithms [27], [28]. However, these algorithms and our proposed approach rely on different clustering criteria, resulting in distinct clustering outcomes and affecting performance.

To assess the effectiveness of our proposed method, we compare its performance against two representative baselines: the Iterative Federated Clustering Algorithm (IFCA) [27] and Clustered Federated Learning: Model-Agnostic Distributed Multi-Task Optimization (MADMO) [28]. Table 5 reports both the clustering results and the corresponding average performance metrics across all contexts. For each value of codeword length, we perform multiple experiments using distinct random seeds and present the resulting clustering outcomes. A notation such as context cluster "(c_1, c_2)" denotes that the CFL algorithm has grouped contexts c_1 and c_2 into the same cluster. For IFCA, which requires the number of clusters to be specified a priori, we fix $K = 4$, and our method infers the number of clusters automatically during training.

The results demonstrate that CFL-GP consistently identifies the context cluster $\{(c_1, c_2), (c_3, c_4), (c_5, c_6), (c_7, c_8)\}$

TABLE 5. Comparison of NMSE performance (dB) and clustering results for the CFL methods.

N_{cl}	Algorithm	NMSE	Generated clusters (context indices)
64-bit	CFL-GP	**-4.94**	$\{(c_1,c_2),(c_3,c_4),(c_5,c_6),(c_7,c_8)\}$
	MADMO	-4.26	$\{(c_1,\ldots,c_8)\}$
	IFCA	-4.29	$\{(c_1,\ldots,c_8)\}$
128-bit	CFL-GP	**-6.60**	$\{(c_1,c_2),(c_3,c_4),(c_5,c_6),(c_7,c_8)\}$
	MADMO	-6.28	$\{(c_1,c_2,c_3,c_4),(c_5,c_6,c_7,c_8)\}$
	IFCA	-5.72	$\{(c_1,\ldots,c_8)\}$
256-bit	CFL-GP	**-8.18**	$\{(c_1,c_2),(c_3,c_4),(c_5,c_6),(c_7,c_8)\}$
	MADMO	-7.38	$\{(c_1,c_2,c_5,c_6),(c_3,c_4,c_7,c_8)\}$, $\{(c_1,\ldots,c_8)\}$
	IFCA	-7.06	$\{(c_1,\ldots,c_8)\}$

TABLE 6. Clustering robustness measured by adjusted rand index (ARI) with varying relative sizes of the local dataset per context.

N_{cl}	1/128	1/32	1/8	1/2	1.0
64-bit	0.862	1.0	1.0	1.0	1.0
128-bit	0.444	1.0	1.0	1.0	1.0
256-bit	0.725	1.0	1.0	1.0	1.0

across all nine experiments. These outcomes suggest that CFL-GP effectively captures underlying similarities in channel characteristics, grouping contexts based on both the number of dominant clusters and the base station type (macro vs. micro).

In contrast, IFCA produces a single global model that fails to capture distributional heterogeneity, leading to suboptimal performance. MADMO, on the other hand, yields three distinct clustering patterns across the experiments, with some trials grouping contexts by cell type, e.g., (c_1,c_2,c_5,c_6) vs. (c_3,c_4,c_7,c_8), and others by propagation condition, e.g., (c_1,c_2,c_3,c_4) vs.(c_5,c_6,c_7,c_8).

Among the three methods, CFL-GP consistently achieves the best NMSE performance. This highlights its robustness in discovering context clusters under distributional uncertainty and its ability to enable effective context grouping, leading to improved compression and reconstruction performance.

a: CLUSTERING ROBUSTNESS

We further evaluate the robustness and consistency of the proposed clustering method with respect to the number of data samples available per context. Table 6 summarizes the results under different relative dataset size ratios, where, for example, a ratio of 1/4 indicates that each local context contains one-quarter of the samples compared to the original setup. To assess clustering stability, we employ the Adjusted Rand Index (ARI), where larger values indicate better alignment, and a value of 1 denotes perfect agreement with the target clustering.

The ARI is measured against the clustering output obtained from the proposed method using the full dataset, i.e., $\{(c_1,c_2),(c_3,c_4),(c_5,c_6),(c_7,c_8)\}$, which corresponds to the best-performing clustering configuration. As shown in Table 6, the ARI remains 1.0 when the dataset size per context is reduced to as low as 1/32 of the original, demonstrating

TABLE 7. Average intra-cluster NMSE gap (Max-min fairness index).

N_{cl}	CFL-GP	IFCA	MADMO	Global	Local
64-bit	0.0373	0.5606	0.5564	0.5571	0.0
128-bit	0.0379	0.4611	0.2186	0.4487	0.0
256-bit	0.0338	0.3508	0.2383	0.3465	0.0

TABLE 8. Comparison of cosine similarity and clustering results for the CFL methods.

N_{cl}	Algorithm	Cosine similarity	Generated clusters (context indices)
64-bit	CFL-GP	**0.9248**	$\{(c_1,c_2),(c_3,c_4),(c_5,c_6),(c_7,c_8)\}$
	MADMO	0.9175	$\{(c_1,c_2,c_3,c_4),(c_5,c_6,c_7,c_8)\}$
	IFCA	0.9167	$\{(c_1,\ldots,c_8)\}$
128-bit	CFL-GP	**0.9512**	$\{(c_1,c_2),(c_3,c_4),(c_5,c_6),(c_7,c_8)\}$
	MADMO	0.9511	$\{(c_1,c_2,c_3,c_4),(c_5,c_6),(c_7,c_8)\}$
	IFCA	0.9414	$\{(c_1,c_2),(c_3,c_4),(c_5,c_6),(c_7,c_8)\}$
256-bit	CFL-GP	**0.9645**	$\{(c_1,c_2),(c_3,c_4),(c_5,c_6),(c_7,c_8)\}$
	MADMO	0.9642	$\{(c_1,c_2,c_3,c_4),(c_5,c_6),(c_7,c_8)\}$
	IFCA	0.9583	$\{(c_1,\ldots,c_8)\}$

that the clustering assignments are preserved even under substantial data reduction. Only in extremely data-scarce regimes (e.g., 1/128 of the dataset, corresponding to fewer than 100 samples per context) does the ARI noticeably degrade. In this case, the ARI drops to 0.444 for the 128-bit setting and to 0.725 for the 256-bit setting, while the 64-bit model still achieves 0.862. This indicates that insufficient data can lead to unstable clustering.

Overall, these results confirm that the proposed method maintains consistent and robust clustering behavior under moderate data reduction, and only exhibits degradation in severely undersampled scenarios.

b: MAX-MIN PERFORMANCE FAIRNESS

To further assess clustering quality, Table 7 reports the max-min fairness index, defined as the performance gap between the best and worst NMSE within each cluster. Ideally, this value should be small, indicating uniform performance across contexts sharing the same decoder. The results confirm that CFL-GP achieves the smallest fairness index among all methods, except for the Local baseline, which trivially yields zero since no contexts are clustered. MADMO attains the next lowest values, consistent with its ability to occasionally form effective clusters, especially where $N_{\text{cl}} = 128$ and 256. The Global model approach, on the other hand, exhibits large fairness gaps due to its inability to differentiate heterogeneous contexts.

c: DIFFERENT LOSS FUNCTIONS

To investigate the adaptability of the proposed context-dependent approach under different loss functions, we consider the cosine similarity loss: $\ell_{\cos}(\mathbf{h}_n,\hat{\mathbf{h}}_n) = 1 - \frac{|\mathbf{h}_n^{\text{H}}\hat{\mathbf{h}}_n|}{\|\mathbf{h}_n\|_2\,\|\hat{\mathbf{h}}_n\|_2}$, where $\mathbf{h}_n \in \mathbb{C}^M$ denotes the channel vector at the n-th subcarrier, and $\hat{\mathbf{h}}_n$ is its reconstruction, which is obtained by FFT of the decoder output. Cosine similarity is directly

related to channel capacity when the decoder output is used for precoding, since maximizing the alignment term $|\mathbf{h}_n^{\mathrm{H}}\hat{\mathbf{h}}_n|$ improves the effective Signal-to-Noise Ratio (SNR). We evaluate the context-dependent approaches using this loss, and report performance and clustering results in Table 8.

The notable observation is that, even under this distinct loss function, the proposed CFL-GP consistently identifies the same clustering pattern as with the NMSE loss, while also achieving the best performance. MADMO occasionally produces the optimal clustering (in terms of the target performance), particularly when $N_{\mathrm{cl}} = 128$, but with less consistency. IFCA yields a single cluster, resulting in suboptimal performance.

VI. LIMITATIONS AND FUTURE DIRECTIONS

Delays in FL: Future work includes addressing heterogeneous CU-BS transmission links that may introduce delays or noise in model and cluster updates. Delays can be mitigated by temporarily excluding late contexts and re-integrating them once updates are available, with cluster assignments adjusted accordingly. Transmission noise can be alleviated through efficient quantization or error-resilient encoding of gradient information over the air. After clustering is completed, which is typically much shorter than the overall training process, the proposed method reduces to conventional FL, where well-studied communication resource allocation strategies-such as energy-efficient scheduling and gradient compression control, e.g., [31] and [32], can be readily applied.

Previously unseen contexts: Another important direction is the handling of entirely new channel distributions that have not been previously observed, for example, due to drastic geographical or environmental changes. Once such a distribution is detected, the system can efficiently initialize from a pre-trained model and update only the corresponding decoder for the newly formed cluster. In rare cases where the distribution is fundamentally different, additional training of the universal encoder may be required.

Advanced model architectures: Future work may also explore more advanced neural architectures, such as attention-based mechanisms [33], to further enhance representation capability. In addition, selectively fine-tuning encoder modules and enabling parameter sharing across decoders may provide greater flexibility for generalization and better adaptation of the proposed context-dependent decoding scheme to diverse channel conditions.

CSI quality and data fidelity: In this work, the experiments assume access to a limited amount of ground-truth local CSI data. Understanding how performance is affected by data fidelity, such as when using synthesized or corrupted CSI, is a worthwhile direction for future exploration.

VII. CONCLUSION

This paper addresses key challenges in data-driven CSI feedback methods, encompassing issues of data heterogeneity, reliance on extensive datasets, and deployment complexity. To tackle these challenges, we introduced a context clustering approach that features distinct decoders for each cluster alongside a universal encoder for all contexts. This system model is trained on CSI data through a CFL algorithm employing gradient similarity for context clustering. Our comprehensive experimentation validated that the proposed model yields superior compression performance, improved sample efficiency, and decreased storage overhead compared to conventional methods, be it a global model or multiple local models. Furthermore, our experimental results emphasized the substantial influence of context clustering results on ultimate performance by showing that the proposed algorithm outperforms representative CFL methods. Overall, our framework and algorithm hold promise for addressing the implementation of data-driven CSI feedback methods in real-world scenarios, especially under constraints like limited data and heterogeneous channel distributions.

APPENDIX A
RELATED WORK

A. NEURAL CSI FEEDBACK

The explicit CSI feedback techniques based on Deep Learning (DL) have surpassed the performance of existing conventional feedback methods, including Compressive sensing-based techniques, as shown in [4] and [7]. These methods often employ an autoencoder architecture tailored for CSI compression and decoding, with its weights optimized using training data drawn from a specific wireless channel distribution.

Research on CSI feedback using the DL-based methods has been focused on improving the structure of neural networks. [4] proposed a DL-based CSI compression method using convolutional layers and fully connected layers, which outperformed conventional compression methods for Frequency Division Duplexing (FDD). Subsequently, tailored neural architectures have been introduced to further improve CSI feedback by leveraging temporal and spatial correlations in CSI [5], [12], [34], [35], [36], [37]. Beyond FDD settings, neural CSI feedback has also been explored for Time Division Duplexing (TDD) systems with multi-user settings [38].

To further enhance flexibility and system-level efficiency, several extended system models have been proposed. Learning-based techniques for joint feedback and channel estimation were introduced in [39] and [40]. A Multi-Task Learning-Based CSI Feedback design, which employs a CSI classification function to select environment-specific decoders, was presented in [41]. Optimization of multi-task learning frameworks via model pre-training was achieved in [42].

Despite the progress achieved by these approaches, they do not fully take into account the constraints, including the challenges posed by the coexistence of diverse channels, the difficulties of distinguishing among channel distributions, and the limitations imposed by having access to only limited datasets. In this paper, we propose a CFL approach that effectively addresses these challenges.

B. FEDERATED LEARNING

FL [13] is a distributed optimization paradigm where local devices, each with their own datasets, train local models, which are subsequently aggregated by averaging their weights. This process of local training and global model aggregation is repeated iteratively until the model converges. FL framework has gained widespread adoption across various fields [43], [44], [45], including wireless communications [46], due to its advantages in communication efficiency, data privacy, and generalization capabilities.

Model Personalization: In real-world FL, non-identical local data distributions degrade the performance of globally trained models. Personalized Federated Learning (PFL) addresses this by adapting global models to local tasks [47], often via transfer learning [48], meta-learning [49], and multitask learning [50].

Federated Learning for Channel Estimation, Prediction, Compression, and Feedback: FL offers a practical solution for deploying DL-based CSI processing by reducing communication overhead for data transmission and preserving user data privacy [24], [51], [52]. In [53], FL with online learning enables BS and UEs to collaboratively learn the hidden sparsity structure in MU-MIMO systems. FL is used to train a global symbol detector across varying DL fading channels in [54] and to support communication-efficient CSI feedback autoencoder training with partial model parameter sharing in [55]. FL is also utilized for channel estimation enabled by federated generative adversarial networks [56].

FL-based frameworks have been extended to multi-rate CSI compression [57], joint CSI estimation and feedback [58], and personalized CSI feedback with quantization for reduced FL training overhead [59]. DL-based CSI prediction under FL in FDD massive MIMO systems is addressed in [60]. In particular, [60] highlights the advantages of FL over centralized training, where transmitting large volumes of high-dimensional CSI data from base stations to a central unit leads to significant communication overhead. FL mitigates this by requiring only the exchange of model parameters, thereby substantially reducing the transmission burden. A hybrid of federated and transfer learning for asynchronous downlink CSI prediction is proposed in [61], while [62] incorporates meta-learning for model initialization with FL-based online fine-tuning. In [63], distributed CSI estimation for cell-free MIMO is achieved via local updates at access points and centralized aggregation with variance reduction.

Key Differences Between Existing CSI Feedback Personalization and Our Approach: In the context of CSI compression, recent works such as [41], [42], and [64] have introduced multitask learning-based personalization strategies. These approaches rely on a shared encoder module with distinct CSI decoders tailored to each local dataset [41], [42], or partial parameter sharing across local models [64]. In [59], fine-tuning of the local model for personalization is achieved after global model training. However, these methods assume prior knowledge of the underlying dataset distribution during training and typically require sufficient volumes of data (e.g., over 40,000 CSI instances per context [41]) or focus predominantly on per-context personalization [41], [59].

In contrast, our work addresses the specific challenge of operating under limited sample availability, e.g., only 10,000 CSI instances per context, while also lacking prior distributional labels and being subject to data privacy constraints. A key aspect of our approach is *the exploration of the benefits of exploiting similarities in channel distributions across diverse contexts*. Rather than assigning a personalized model to each context, we propose a novel strategy that identifies and leverages these distributional similarities to group contexts with similar characteristics. This allows for collaborative training of a shared model, improving efficiency and generalization across contexts. Contexts with substantially different distributions are excluded from these groups to maintain model performance. To realize this, we use our CFL algorithm, CFL-GP, which dynamically groups contexts based on gradient information, optimizing model training for varied and unknown distributional contexts.

Clustered Federated Learning: CFL extends the personalization capabilities of FL by training a separate model for each group of contexts, rather than a single global model for all participants. CFL algorithms aim to identify clusters of contexts with similar data distributions and train one model per cluster. In [28], a bipartition-based CFL algorithm was proposed, wherein contexts are split into groups based on gradient differences from the converged global model. This process is applied recursively to form multiple clusters, each training its own model. Loss-based clustering approaches have also been explored [27], [66], with [27] introducing an efficient CFL algorithm that reduces the computational load by offloading clustering tasks to local contexts, which select the best-performing model from a pool of candidates.

However, existing CFL algorithms often rely on sensitive clustering parameters and assume a large number of contexts for convergence [27], assumptions that may not hold in wireless networks where the number of contexts is limited. Additionally, some of the existing CFL approaches rely on predefined parameters related to gradient statistics for context clustering [28], which may be challenging due to unknown gradient statistics and the variability in channel distributions. Our proposed CFL algorithm is specifically designed for environments with limited contexts, without the need for sensitive hyperparameter tuning, making it more applicable to real-world scenarios.

APPENDIX B
AUTOENCODER ARCHITECTURE

The autoencoder architecture is motivated by inception blocks, with both the encoder and decoder utilizing the inception-style design introduced in [65] and further developed for CSI processing modules [12], [67]. In particular, CRNet [12] modules are employed for angular-delay domain CSI processing, providing lightweight and effective feature extraction, while the VQ-VAE framework is incorporated

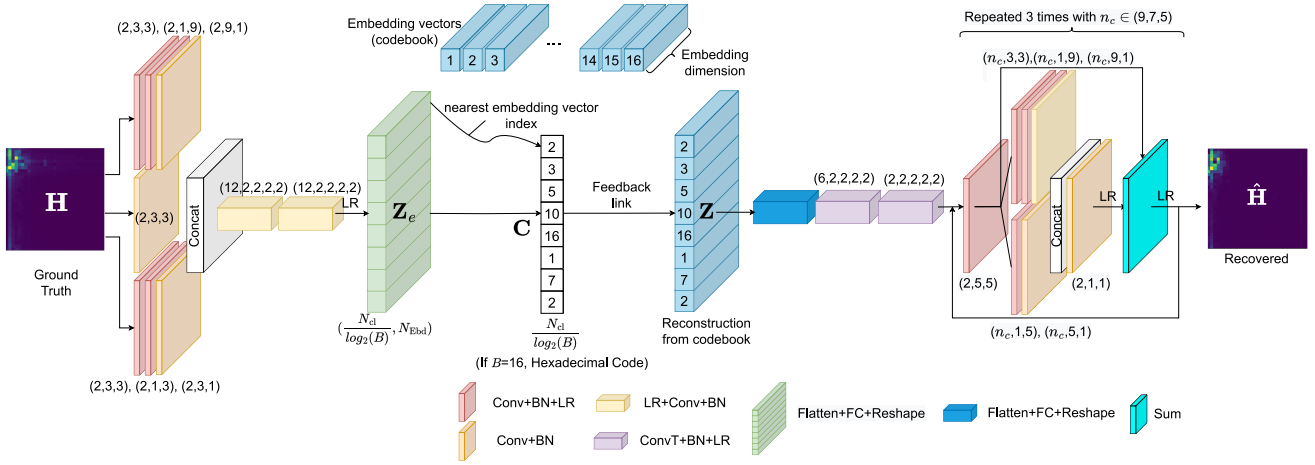

FIGURE 8. The proposed autoencoder architecture. It combines the VQ-VAE [11] for quantization and the Inception modules [65] for CSI encoding and decoding. Further details can be found in [9].

to enable codeword discretization. The combined architecture enables end-to-end gradient-based optimization for the encoder, quantizer, and decoder, forming the VQCINet architecture as depicted in Figure 8. In each block containing convolutional (Conv) or transposed convolutional (ConvT) layers, the numbers in parentheses in Figure 8 correspond to (output channels, kernel width, kernel height). If five numbers are listed, the final two indicate the stride sizes for width and height, respectively.

We consider N_{cl}-bit compression of a CSI instance $\mathbf{H}$. To achieve this, we introduce a base B where $B > 2$. This allows the codeword length to be $N_{\mathrm{cl}}/\log_2(B)$, while preserving the same cardinality as the N_{cl}-bit binary codeword. This approach leads to a more compact model, as directly generating binary outputs from the encoder would necessitate a considerably larger encoder with a higher parameter count. The overall architecture follows an Encoder-Quantizer-Decoder structure.

Encoder: The encoder receives a cropped CSI matrix in the angular-delay domain $\mathbf{H}$, which is processed through three parallel paths. Each path contains a convolutional layer (Conv), batch normalization (BN), and leaky ReLU (LR) activation. The outputs of these paths are concatenated along the channel dimension and processed by two additional LR+Conv+BN blocks. The resulting data is vectorized and passed through a Fully Connected (FC) layer to obtain a vector of size $\frac{N_{\mathrm{cl}}N_{\mathrm{emb}}}{\log_2(B)}$, where B is the base of the codeword representation and N_{emb} is the embedding dimension. This vector is reshaped into a matrix $\frac{N_{\mathrm{cl}}}{\log_2(B)} \times N_{\mathrm{emb}}$, referred to as $\mathbf{Z}_e$. This representation is then quantized.

Quantizer: The representation $\mathbf{Z}_e$ is quantized using a codebook consisting of B distinct vectors, each of dimension N_{emb}. The encoder output can be viewed as a collection of $N_{\mathrm{cl}}/\log_2(B)$ embedding vectors, each of size N_{emb}. Each vector in $\mathbf{Z}_e$ is mapped to the closest embedding vector in the codebook based on the smallest l_2 norm. This process yields the quantized output $\mathbf{Z}$, as illustrated in Figure 8 in Appendix B. The quantization achieves N_{cl}-bit compression by selecting from B distinct embedding vectors, where the length is $N_{\mathrm{cl}}/\log_2(B)$. We ensure that B is chosen such that $N_{\mathrm{cl}}/\log_2(B)$ is integer valued.

For CSI feedback, the encoder transmits the *indices of the selected codebook vectors* over the wireless feedback link to the BS. The sequence of indices corresponding to the embedding vectors forms the codeword $\mathbf{C}$.

Decoder: At the BS, the received codeword $\mathbf{C}$ is mapped to an embedding tensor and reconstructed into CSI $\hat{\mathbf{H}}$. The decoder consists of three stages:

1) *Upsampling:* Two ConvT layers progressively expand the embedding. The first layer maps $12 \rightarrow 6$ channels with kernel 2×2, stride 2, and no bias; the second maps $6 \rightarrow 2$ channels with the same kernel and stride. Each ConvT is followed by a BN and an LR with a slope of 0.3.

2) *Feature refinement:* The upsampled tensor is refined by three stacked CR blocks adapted from CRNet [12]. In each CR block, two parallel paths are used: Path 1 applies 3×3, 1×9, and 9×1 Conv layers to capture correlations across delay and angular dimensions; Path 2 applies 1×5 and 5×1 Conv layers for medium-range dependencies. All Conv layers are followed by BN and LR. The outputs of the two paths are concatenated, reduced back to 2 channels via a 1×1 Conv, and merged with the block input through a residual connection, followed by LR. The three CR blocks are stacked sequentially with intermediate channel sizes of 9, 7, and 5, respectively.

3) *Output:* A final 1×1 Conv followed by a Sigmoid activation produces the reconstructed CSI $\hat{\mathbf{H}}$.

The autoencoder architecture presented in Section VII-B is not inherently differentiable due to the quantization step applied to the encoder's output. To enable gradient-based end-to-end training, we employ the *stop-gradient* operator [11]. Specifically, we modify the forward computation of the encoder's output as $\mathbf{Z} = \mathbf{Z}_e + \mathtt{sg}[\mathbf{Z} - \mathbf{Z}_e]$ where

$\mathtt{sg}$ represents the stop-gradient operator. This operator treats its input as a constant, effectively blocking the gradient propagation through the input. This formulation ensures that the gradient backpropagation of the quantization process is skipped, and the gradient of the MSE loss with respect to the encoder representation $\mathbf{Z}_e$ is replaced by the gradient with respect to the quantized representation $\mathbf{Z}$. We also use the VQ-VAE loss function ℓ' instead of ℓ, which is represented as

$$\ell'(\mathbf{H}, \hat{\mathbf{H}}) = \ell(\mathbf{H}, \hat{\mathbf{H}}) + \|\mathtt{sg}[\mathbf{Z}_e] - \mathbf{Z}\|^2 + \|\mathbf{Z}_e - \mathtt{sg}[\mathbf{Z}]\|^2. \tag{6}$$

The FLOP complexity of the autoencoder is 5.58M, 5.68M, and 5.88M for the 64-bit, 128-bit, and 256-bit settings, respectively.

ACKNOWLEDGMENT

An earlier version of this paper was presented in part at the 41st International Conference on Machine Learning (ICML 2024) [1]. Code and dataset are available at https://github.com/Heasung-Kim/clustered-federated-learning-to-support-context-dependent-csi-decoding

REFERENCES

[1] H. Kim, H. Kim, and G. De Veciana, "Clustered federated learning via gradient-based partitioning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Apr. 2024, pp. 24137–24193.

[2] E. Dahlman, S. Parkvall, and J. Skold, *4G: LTE/LTE-Advanced for Mobile Broadband*. New York, NY, USA: Academic, 2013.

[3] A. Zaidi, F. Athley, J. Medbo, U. Gustavsson, G. Durisi, and X. Chen, *5G Physical Layer: Principles, Models and Technology Components*. New York, NY, USA: Academic, 2018.

[4] C.-K. Wen, W.-T. Shih, and S. Jin, "Deep learning for massive MIMO CSI feedback," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 748–751, Oct. 2018.

[5] T. Wang, C.-K. Wen, S. Jin, and G. Y. Li, "Deep learning-based CSI feedback approach for time-varying massive MIMO channels," *IEEE Wireless Commun. Lett.*, vol. 8, no. 2, pp. 416–419, Apr. 2019.

[6] J. Guo, C.-K. Wen, S. Jin, and G. Y. Li, "Convolutional neural network-based multiple-rate compressive sensing for massive MIMO CSI feedback: Design, simulation, and analysis," *IEEE Trans. Wireless Commun.*, vol. 19, no. 4, pp. 2827–2840, Apr. 2020.

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

[8] X. Lin, "An overview of 5G advanced evolution in 3GPP release 18," *IEEE Commun. Standards Mag.*, vol. 6, no. 3, pp. 77–83, Sep. 2022.

[9] H. Kim. (2025). *Github Repository for Clustered Federated Learning to Support Context-Dependent CSI Decoding*. [Online]. Available: https://github.com/Heasung-Kim/clustered-federated-learning-to-support-context-dependent-csi-decoding

[10] B. Wang, F. Gao, S. Jin, H. Lin, and G. Y. Li, "Spatial- and frequency-wideband effects in millimeter-wave massive MIMO systems," *IEEE Trans. Signal Process.*, vol. 66, no. 13, pp. 3393–3406, Jul. 2018.

[11] A. Van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 6309–6318.

[12] Z. Lu, J. Wang, and J. Song, "Multi-resolution CSI feedback with deep learning in massive MIMO system," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.

[13] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, vol. 54, A. Singh and J. Zhu. Eds., Fort Lauderdale, FL, USA, 2017, pp. 1273–1282.

[14] J. Choi, D. J. Love, and P. Bidigare, "Downlink training techniques for FDD massive MIMO systems: Open-loop and closed-loop training with memory," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 802–814, Oct. 2014.

[15] M. B. Mashhadi and D. Gündüz, "Deep learning for massive MIMO channel state acquisition and feedback," *J. Indian Inst. Sci.*, vol. 100, no. 2, pp. 369–382, Apr. 2020.

[16] T. Lee, J. Park, H. Kim, and J. G. Andrews, "Generating high dimensional user-specific wireless channels using diffusion models," *IEEE Trans. Wireless Commun.*, early access, Jul. 26, 2025, doi: 10.1109/TWC.2025.3600286.

[17] X. Lin, "Artificial intelligence in 3GPP 5G-advanced: A survey," 2023, *arXiv:2305.05092*.

[18] X. Lin, "An overview of AI in 3GPP RAN release 18: Enhancing next-generation connectivity?" *Global Commun.*, Mar. 2024.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[20] M. Löffler, A. Y. Zhang, and H. H. Zhou, "Optimality of spectral clustering in the Gaussian mixture model," *Ann. Statist.*, vol. 49, no. 5, pp. 2506–2530, Oct. 2021.

[21] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, Nov. 1987.

[22] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, pp. 2825–2830, 2012.

[23] K. Pillutla, K. Malik, A.-R. Mohamed, M. Rabbat, M. Sanjabi, and L. Xiao, "Federated learning with partial model personalization," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 17716–17758.

[24] J. Guo, C.-K. Wen, S. Jin, and X. Li, "AI for CSI feedback enhancement in 5G-advanced," *IEEE Wireless Commun.*, vol. 31, no. 3, pp. 169–176, Jun. 2024.

[25] Z. Du, Z. Liu, H. Li, S. Fan, X. Gu, and L. Zhang, "Dig-CSI: A distributed and generative model assisted CSI feedback training framework," *IEEE Wireless Commun. Lett.*, vol. 13, no. 8, pp. 2035–2039, Aug. 2024.

[26] X. Lin, "The bridge toward 6G: 5G-advanced evolution in 3GPP release I9," *IEEE Commun. Standards Mag.*, vol. 9, no. 1, pp. 28–35, Mar. 2025.

[27] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2020, pp. 19586–19597.

[28] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3710–3722, Aug. 2021.

[29] S. Jaeckel, L. Raschkowski, K. Börner, and L. Thiele, "QuaDRiGa: A 3-D multi-cell channel model with time evolution for enabling virtual field trials," *IEEE Trans. Antennas Propag.*, vol. 62, no. 6, pp. 3242–3256, Jun. 2014.

[30] S. Jaeckel et al., "QuaDRiGa—Quasi deterministic radio channel generator, user manual and documentation," Fraunhofer Heinrich Hertz Inst., Tech. Rep. v2.6.1, Jul. 2021.

[31] C. T. Dinh et al., "Federated learning over wireless networks: Convergence analysis and resource allocation," *IEEE/ACM Trans. Netw.*, vol. 29, no. 1, pp. 398–409, Feb. 2021.

[32] P. Hegde, G. D. Veciana, and A. Mokhtari, "Network adaptive federated learning: Congestion and lossy compression," in *Proc. IEEE INFOCOM*, Apr. 2023, pp. 1–10.

[33] Y. Cui, A. Guo, and C. Song, "TransNet: Full attention network for CSI feedback in FDD massive MIMO system," *IEEE Wireless Commun. Lett.*, vol. 11, no. 5, pp. 903–907, May 2022.

[34] X. Li and H. Wu, "Spatio-temporal representation with deep neural recurrent network in MIMO CSI feedback," *IEEE Wireless Commun. Lett.*, vol. 9, no. 5, pp. 653–657, May 2020.

[35] Q. Li, A. Zhang, P. Liu, J. Li, and C. Li, "A novel CSI feedback approach for massive MIMO using LSTM-attention CNN," *IEEE Access*, vol. 8, pp. 7295–7302, 2020.

[36] Y. Liu and O. Simeone, "HyperRNN: Deep learning-aided downlink CSI acquisition via partial channel reciprocity for FDD massive MIMO," in *Proc. IEEE 22nd Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Sep. 2021, pp. 31–35.

[37] Q. Cai, C. Dong, and K. Niu, "Attention model for massive MIMO CSI compression feedback and recovery," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2019, pp. 1–5.

[38] J. Park, F. Sohrabi, A. Ghosh, and J. G. Andrews, "End-to-end deep learning for TDD MIMO systems in the 6G upper midbands," *IEEE Trans. Wireless Commun.*, vol. 24, no. 3, pp. 2110–2125, Mar. 2024.
[39] Y. Sun, W. Xu, L. Fan, G. Y. Li, and G. K. Karagiannidis, "AnciNet: An efficient deep learning approach for feedback compression of estimated CSI in massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 9, no. 12, pp. 2192–2196, Dec. 2020.
[40] J. Guo, C.-K. Wen, and S. Jin, "CANet: Uplink-aided downlink channel acquisition in FDD massive MIMO using deep learning," *IEEE Trans. Commun.*, vol. 70, no. 1, pp. 199–214, Jan. 2022.
[41] X. Li, J. Guo, C.-K. Wen, S. Jin, S. Han, and X. Wang, "Multi-task learning-based CSI feedback design in multiple scenarios," *IEEE Trans. Commun.*, vol. 71, no. 12, pp. 7039–7055, Dec. 2023.
[42] B. Zhang, H. Li, X. Liang, X. Gu, and L. Zhang, "Multi-task training approach for CSI feedback in massive MIMO systems," *IEEE Commun. Lett.*, vol. 27, no. 1, pp. 200–204, Jan. 2023.
[43] K. Bonawitz et al., "Towards federated learning at scale: System design," in *Proc. MLSys*, vol. 1, Apr. 2019, pp. 374–388.
[44] L. Yang, B. Tan, V. W. Zheng, K. Chen, and Q. Yang, "Federated recommendation systems," in *Federated Learning*. Cham, Switzerland: Springer, 2020, pp. 225–239.
[45] N. Rieke et al., "The future of digital health with federated learning," *NPJ Digit. Med.*, vol. 3, no. 1, p. 119, 2020.
[46] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities, and challenges," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 46–51, Jun. 2020.
[47] V. Kulkarni, M. Kulkarni, and A. Pant, "Survey of personalization techniques for federated learning," in *Proc. 4th World Conf. Smart Trends Syst., Secur. Sustainability (WorldS4)*, Jul. 2020, pp. 794–797.
[48] K. Wang, R. Mathews, C. Kiddon, H. Eichner, F. Beaufays, and D. Ramage, "Federated evaluation of on-device personalization," 2019, *arXiv:1910.10252*.
[49] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," 2020, *arXiv:2002.07948*.
[50] O. Marfoq, G. Neglia, A. Bellet, L. Kameni, and R. Vidal, "Federated multi-task learning under a mixture of distributions," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2021, pp. 15434–15447.
[51] J. Wang, X. Zhang, Z. Lu, J. Tan, and H. Zhang, "Practical deployment for deep learning-based CSI feedback systems: Generalization challenges and enabling techniques," *IEEE Commun. Mag.*, early access, Mar. 12, 2025, doi: 10.1109/MCOM.001.2400305.
[52] V. A. Nugroho and B. M. Lee, "A survey of federated learning for mmWave massive MIMO," *IEEE Internet Things J.*, vol. 11, no. 16, pp. 27167–27183, Aug. 2024.
[53] X. Zheng and V. Lau, "Federated online deep learning for CSIT and CSIR estimation of FDD multi-user massive MIMO systems," *IEEE Trans. Signal Process.*, vol. 70, pp. 2253–2266, 2022.
[54] M. B. Mashhadi, N. Shlezinger, Y. C. Eldar, and D. Gunduz, "Fedrec: Federated learning of universal receivers over fading channels," in *Proc. IEEE Stat. Signal Process. Workshop (SSP)*, Jul. 2021, pp. 576–580.
[55] Z. Du, H. Li, L. Li, B. Zhang, Z. Liu, and X. Gu, "Training CSI feedback model with federated learning in massive MIMO systems," in *Proc. 8th IEEE Int. Conf. Netw. Intell. Digit. Content (IC-NIDC)*, Nov. 2023, pp. 446–450.
[56] Y. Guo, Z. Qin, X. Tao, and O. A. Dobre, "Federated generative-adversarial-network-enabled channel estimation," *Intell. Comput.*, vol. 3, Jan. 2024, Art. no. 0066.
[57] B. Hu, H. Xu, Z. Wang, L. Zhao, and A. Zhou, "Multiple-compression-rate CSI feedback for mm-wave massive MIMO based on federated learning," *Phys. Commun.*, vol. 63, Apr. 2024, Art. no. 102305.
[58] L. Zhao, H. Xu, Z. Wang, X. Chen, and A. Zhou, "Joint channel estimation and feedback for mm-wave system using federated learning," *IEEE Commun. Lett.*, vol. 26, no. 8, pp. 1819–1823, Aug. 2022.
[59] Y. Cui, J. Guo, C.-K. Wen, and S. Jin, "Communication-efficient personalized federated edge learning for massive MIMO CSI feedback," *IEEE Trans. Wireless Commun.*, vol. 23, no. 7, pp. 7362–7375, Jul. 2024.
[60] W. Hou et al., "Federated learning for DL-CSI prediction in FDD massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 10, no. 8, pp. 1810–1814, Aug. 2021.
[61] J. Sun, Y. Zhang, G. Gui, H. Zhao, H. Gacanin, and H. Sari, "Interacting federated and transfer learning-aided CSI prediction for intelligent cellular networks," *IEEE Trans. Veh. Technol.*, vol. 72, no. 12, pp. 15776–15787, Dec. 2023.
[62] H. Zhao, W. Liu, W. Xia, Y. Shen, and H. Zhu, "Enhanced meta-transfer learning assisted CSI feedback in massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 14, no. 2, pp. 499–503, Feb. 2025.
[63] Y. Xu and S. Wang, "Efficient federated learning algorithm design for distributed channel estimation in cell-free massive MIMO systems," *IEEE Trans. Commun.*, vol. 73, no. 9, pp. 7899–7913, Sep. 2025.
[64] S. M. Bathala, S. Amuru, and K. K. Kuchi, "Multi-task learning for massive MIMO CSI feedback," in *Proc. 3rd Int. Conf. Artif. Intell. Mach. Learn. Syst.*, Oct. 2023, pp. 1–4.
[65] C. Szegedy et al., "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
[66] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," 2020, *arXiv:2002.10619*.
[67] Y. Xu, M. Zhao, S. Zhang, and H. Jin, "DFECsiNet: Exploiting diverse channel features for massive MIMO CSI feedback," in *Proc. 13th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2021, pp. 1–5.

HEASUNG KIM (Member, IEEE) received the B.S. degree in computer and communication engineering from Korea University, Seoul, South Korea, in 2017, and the M.S. degree from the Department of Electrical and Computer Engineering, Seoul National University, Seoul, in 2019. He is currently pursuing the Ph.D. degree in electrical and computer engineering with The University of Texas at Austin. From 2019 to 2021, he was a Machine Learning Engineer with Samsung Network Business and Samsung Electronics. His research interests include wireless networks, information theory, and machine learning algorithms.

HYEJI KIM (Senior Member, IEEE) received the B.S. degree in electrical engineering from KAIST in 2011 and the Ph.D. degree in electrical engineering from Stanford University in 2016. She is currently an Assistant Professor with the Department of Electrical and Computer Engineering at The University of Texas at Austin. Before joining UT Austin, she was a Post-Doctoral Research Associate at the University of Illinois at Urbana-Champaign and a Researcher at Samsung AI Research Cambridge, U.K. Her research interests include the intersection of information theory, machine learning, and wireless communications. She was a recipient of the NSF CAREER Award.

GUSTAVO DE VECIANA (Fellow, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of California, Berkeley, in 1987, 1990, and 1993, respectively. He is currently a Professor and the Associate Chair of the Department of Electrical and Computer Engineering. He was the Director and the Associate Director of the Wireless Networking and Communications Group (WNCG), University of Texas at Austin, from 2003 to 2007. His research interests include the design, analysis, and control networks, information theory, and applied probability. His current research interests include measurement, modeling, and performance evaluation; wireless and sensor networks; and architectures and algorithms to design reliable computing and network systems. In 2009, he was designated as an IEEE Fellow for his contributions to the analysis and design of communication networks. He was a recipient of the National Science Foundation CAREER Award in 1996 and a co-recipient of the six best paper awards, including the IEEE William McCalla Best ICCAD Paper Award in 2000 and the Best Paper Award in *ACM Transactions on Design Automation of Electronic Systems* in January 2002 and January 2004. He has been an Editor for IEEE/ACM Transactions on Networking.