Variable Code Size Autoencoder (VCSA) Meets CSI Compression in Model Generalization

Yifei Song‡, Juan Roa†, Renjian Zhao†, Zhigang Rong†, Weimin Xiao†, Jalal Jalali†, and Baoling Sheen†
†Wireless Research and Standards, Futurewei Technologies, Bridgewater, NJ 08807, USA
‡ Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24060 USA
Email:{jroa,rzhao,zrong,weimin.xiao,jfaghih,bsheen}@futurewei.com and yifeisong@vt.edu

***Abstract*—To fully exploit the advantages of spatial multiplexing gains in Multiple-Input Multiple-Output (MIMO) systems operating in Frequency Division Duplex (FDD) mode, it is crucial to develop a robust Channel State Information (CSI) feedback compression and reconstruction strategy. This strategy should effectively reduce the communication overhead while maintaining close-to-optimal reconstruction accuracy. However, conventional codebook-based approaches, as specified in 3rd Generation Partnership Project (3GPP) standards and typical deep learning (DL)-based techniques, encounter challenges like high air-interface overhead and/or poor generalization performance across scenarios. In this paper, we introduce a novel neural network architecture named Variable Code Size Autoencoder (VCSA), a unified two-sided model that not only enables the user equipment (UE) to generate variable-size encoder output but also allows the network to take variable-size input to reconstruct the CSI. Empirical results show that leveraging VCSA together with quantization can achieve comparable performance as using multiple encoder output size-specific autoencoders. Additionally, we show that with proper training strategy, the VCSA model achieves decent generalization performance between urban macro (UMa) and urban micro (UMi) scenarios, which further demonstrates the benefits of using DL techniques to provide practical solutions in fifth-generation (5G) networks and beyond.**

***Index Terms*—CSI Compression, 3rd Generation Partnership Project (3GPP), Model Generalization, Autoencoder (AE), Deep Learning (DL), and MIMO**

I. INTRODUCTION

MIMO stands as a promising technique aimed at enhancing spectrum and energy efficiency in the context of next-generation wireless system [1], [2]. However, this advancement introduces a new challenge, particularly with regard to base stations (BS). The next-generation BS (gNB) must acquire real-time CSI for precoding purposes, a requirement accentuated in FDD systems. Downlink CSI acquisition involves two primary steps. First, the UE estimates the downlink CSI by utilizing the received pilot signals transmitted by the BS. Subsequently, the UE relays this estimated downlink CSI to the BS via the uplink control channel. In the context of massive MIMO systems, where the BS is equipped with a large number of antennas, the resulting CSI dimension becomes extensive, necessitating significant feedback overhead.

Conventional CSI feedback methods in existing 5G system depend on codebooks, as reviewed in [3], often face challenges in finding the optimal trade-off between computational complexity and accuracy. On the other hand, as suggested by signal processing researchers, compressive sensing techniques, as discussed in [4] and [5], necessitate ideal assumptions such as channel sparsity, which may not always hold true in practical systems. In recent years, Machine Learning (ML), DL in particular, based approach has been leveraged in wireless communication to enhance conventional communication functionalities like channel estimation [6], precoding/beamforming in massive MIMO [7], signal detection [8], to name a few. For CSI feedback compression and reconstruction, 3GPP Release 18 has set up a study item (abbreviated as NR_AIML_Air in [9]) to explore the potential of Artificial Intelligence (AI)/ML-based solutions for three identified use cases, and CSI feedback enhancement is one of the representative use cases. For DL-based CSI feedback compression, the common architecture adopted the idea of the AE used in image compression [10]. Even though these approaches achieved promising performance in reconstruction accuracy, they are all based on a common assumption that the encoder output and decoder input share the same shapes. In real deployment scenarios, the encoder may reside on the UE, and the decoder may reside on the BS. Thus, this constraint poses challenges for UEs and network vendors who may employ their own strategies in what encoder output and/or decoder input shapes to be supported. Additionally, the eventual CSI feedback overhead or payload depends not only on the size of the encoder output or latent space, which is also referred to as code size interchangeably in this article but also on the quantization scheme applied. It is apparent to understand that developing a versatile model that can adapt to various CSI feedback payloads is a very important part of the overall AI/ML-based solution.

As discussed earlier, previous works have demonstrated the effectiveness of DL-based methods for CSI compression, resulting in improved compression ratios and reconstruction accuracy [11]–[14]. Some related literature [15]–[17] employs similar terminology, but the problem spaces addressed are different. A conventional approach to address variable code sizes would involve developing multiple models, one model for each code size. However, this approach demands significant resources to manage and store several well-trained models and the associated model switching strategy. Recently, authors in [18] introduce a feedback overhead control unit (FOCU) for adaptive CSI feedback encoding. It accommodates varying

979-8-3503-7099-7/24/$31.00 ©2024 IEEE

feedback bit rates, optimizing storage. However, different feedback bit rates may require different configurations. In this case, multiple AEs would have to be offloaded to each UE, resulting in increased storage requirements at the user side. Our design involves a similar idea to the FOCU but using data augmentation to achieve efficient use of the training data independently of the quantization block. This provides flexible CSI feedback encoding with a single encoder. Meanwhile, [19] employs Nested Dropout (ND) to ensure decoder robustness to feedback size variations, though it complicates training. In contrast, we preprocess data for variable feedback sizes and leverage data augmentation for efficient training.

The main contributions of this paper are as follows.

- We present a unified AE Neural Network (NN) architecture, named as Variable Code Size Autoencoder (VCSA), that can adapt to multiple code sizes. The goal is to mitigate the complexities and effort associated with developing and managing multiple models to support multiple code sizes. The NN is then trained using datasets from multiple deployment scenarios. The trained unified model for CSI compression and reconstruction has the ability not only to adapt to different CSI code sizes but also to generalize across multiple scenarios.
- We employ a renewable quantization approach tailored for variable CSI payload sizes and evaluate its ability to generalize across various scenarios. It's important to note that generalization across scenarios poses a general DL challenge when the training and inference data distributions differ. Our objective is to develop a model that can achieve performance levels comparable to those of multiple scenario-specific models, and each specifically designed for a single code size and/or scenario while keeping the model complexity at a reasonable level.

The remainder of this paper is organized as follows: In Section II, we introduce the wireless system model and the details of our proposed DL model. Section III provides a comprehensive overview of the proposed experiment settings and discusses the evaluation procedures and results. Finally, Section IV concludes the paper.

II. System Model

A. Wireless System Model

In this paper, we consider a multiple-cell downlink cellular FDD scheme employed in a MIMO system. The system consists of N_t antennas at the BS and N_r antennas at the user equipment UE, where both N_t and N_r are greater than or equal to 1. The system utilizes Orthogonal Frequency Division Multiplexing (OFDM) with S subbands, where each subband consists of C resource blocks (RBs). The downlink channel is expressed as:

$$\boldsymbol{H} = \big[\boldsymbol{H}_1, \boldsymbol{H}_2, \boldsymbol{H}_3, \ldots, \boldsymbol{H}_S\big], \tag{1}$$

where $\boldsymbol{H}_s \in \mathbb{C}^{N_r \times N_t}$, $1 \leq s \leq S$, is the s-th subband of the downlink channel. This channel is utilized by employing single-stream downlink transmission and ideal channel estimation at the UE side. The eigenvector for the s-th subband, represented as $\boldsymbol{p}_s \in \mathbb{C}^{N_t \times 1}$ with normalization $\mathrm{tr}(\boldsymbol{p}_s \boldsymbol{p}_s^H) = 1$, is directly employed as the downlink precoding vector. This vector can be calculated using eigenvector decomposition in the following equation:

$$\boldsymbol{H}_s^H \boldsymbol{H}_s \boldsymbol{p}_s = \lambda_s \boldsymbol{p}_s. \tag{2}$$

The symbol λ_s represents the largest eigenvalue of the matrix $\boldsymbol{H}_s^H \boldsymbol{H}_s$, which also signifies the precoding power gain achieved from a MIMO system. It's essential to mention that all S eigenvectors must be sent to the BS. This helps create the downlink precoding beamforming for MU-MIMO UEs, particularly for methods like the Zero-forcing algorithm that uses them directly as input. Therefore, a total of $S \times N_t$ complex coefficients should be compressed and recovered for each channel sample using various compression techniques. In this work, the Squared Generalized Cosine Similarity (SGCS), an intermediate Key Performance Indicator (KPI) agreed in 3GPP Rel 18 [20], on the s-th subband, denoted as SGCS_s, is used to assess the accuracy of CSI reconstruction.

$$\mathrm{SGCS}_s = \left(\frac{|\boldsymbol{p}_s^H \boldsymbol{p}_s'|}{||\boldsymbol{p}_s|| ||\boldsymbol{p}_s'||} \right)^2. \tag{3}$$

In the above equation, $\boldsymbol{p}_s'$ represents the recovered eigenvector for the s-th subband. The average SGCS across all subbands is denoted as $\mathrm{SGCS} = \frac{1}{S} \sum_{s=1}^{S} \mathrm{SGCS}_s$, where SGCS is ≤ 1. A higher SGCS value signifies more accurate CSI reconstruction.

B. Variable Code Size Autoencoder (VCSA)

A generalized model introduced in this paper is an AE model which has a common encoder model from the UE side that can generate different sizes of encoder outputs, referred as variable code sizes. At the same time, a common decoder at the BS is used to reconstruct the CSI from variable sizes of the encoder output received. Fig. 1 depicts the high-level architecture design. The CSI generation part represents the CSI compression at the UE side. After the UE receives the CSI reference signal and performs the estimation, the estimated CSI matrix is pre-processed as channel eigenvectors to reduce the amount of feedback overhead. In this paper, we assume max rank is 1. As shown in the CSI generation part, the CSI eigenvector is first compressed by an encoder followed by applying different masks to adjust the encoder output sizes based on the configuration. In this work, we use two types of masks, where mask 1 represents removing the second half of the codes in the encoder output while mask 2 keeps all the codes in the encoder output.

The encoded output is then quantized using an add-on scalar quantizer, which is learned from the encoder outputs from training samples with mask 2 using K-Means, an unsupervised clustering algorithm. As a result, the over-the-air payload size depends on the code size (type of masks being applied) and quantization bits per code which is log_2(number of clusters). The payload is calculated as payload (bits) = code size $\times$ quantization bits. In this paper, we adopt code sizes of 16 and 32 with $[4, 5, 8]$ bits per code, which result into $[80$ bits, 128 bits, 256 bits$]$ according to CSI payload size categories agreed in 3GPP Rel 18:

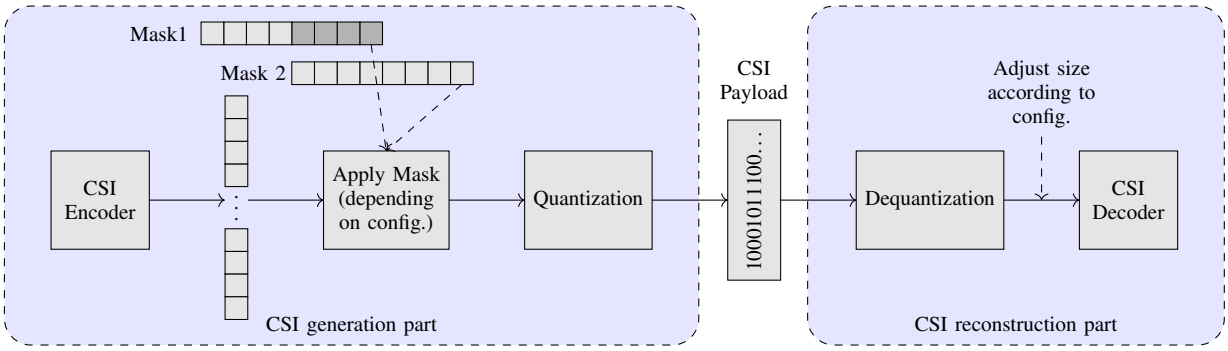

Fig. 1: Generalized Model Architecture with Multiple Code Sizes.

small (payload $\leq$ 80 bits), medium (100 bits $\leq$ payload $\leq$ 140 bits), and large (payload $\geq$ 230 bits). Note that we assume BS and UE have common knowledge of the code size(s) and quantization bits per code prior to CSI compression and reconstruction procedure. Additionally, we assume perfect channel estimation at the UE side, along with perfect CSI feedback from the UE to the BS. At the CSI reconstruction part, the BS first uses the same quantization dictionary to convert the binary payloads to the corresponding floating-point codes (cluster centroids). Depending on the configuration, if mask 1 was applied, the BS pads zeros to the end of the converted codes where the size of zeros equals the size of the codes. Otherwise, the codes will not be modified. The adjusted codes are then sent to the CSI decoder to reconstruct the eigenvectors.

III. Performance Evaluation

A. Experimental Setup

In this section, we discuss the performance evaluation procedures and results of the VCSA and its generalization capabilities across deployment scenarios within the context of the 5G OFDM system. The performance assessment is carried out using two distinct datasets corresponding to dense urban scenarios, denoted as UMa and UMi based on the specifications delineated in the 3GPP 38.901 channel model [21], as detailed in Table I. The setup involves 32 transmit antennas at the gNB (base station), and 4 receiving antennas at the UE. The transmitter power is configured at 44 dBm for a 20 MHz bandwidth and the arrangement of transmitter antennas follows a pattern of $(8, 8, 2, 1, 1, 2, 8)$, with corresponding horizontal and vertical spacing set at 0.5λ and 0.8λ respectively. The configuration of receiver antennas adopts a pattern of $(1, 2, 2, 1, 1, 1, 2)$, with both horizontal and vertical spacing maintained at 0.5λ. The operating frequency is centered around 4 GHz within the Frequency Range 1 (FR1) spectrum, with a sub-carrier spacing of 30 kHz. In our system-level simulation (SLS), we split the bandwidth into 12 sub-bands, each comprising 8 resource blocks.

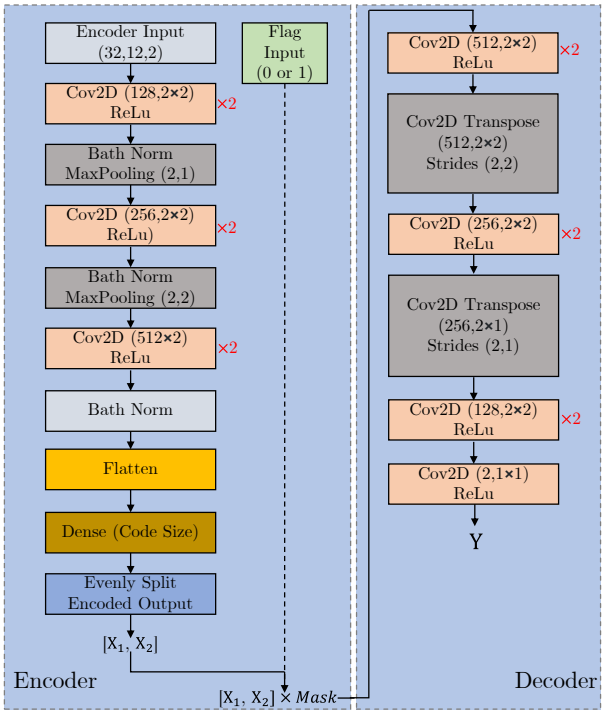

Fig. 2: VCSA Neural Network Structure.

TABLE I: Data Details.

Parameter	Value
Duplex, Waveform	FDD, OFDM
Scenario	Dense Urban (UMa/UMi)
Channel Model	3GPP 38.901 (3D Channel Model)
Frequency Range	FR1 only, 4GHz.
Number of Tx Antenna	32
Tx Antenna Setup and Layouts	$(8, 8, 2, 1, 1, 2, 8)$ (dH,dV) $= (0.5, 0.8)\lambda$
Number of Rx Antenna	4
UE Antenna Setup and Layouts	$(1, 2, 2, 1, 1, 1, 2)$ (dH,dV) $= (0.5, 0.5)\lambda$ (rank 1 only)
BS Tx power	44dBm for 20MHz
Numerology: SCS	30kHz for 4GHz
UE Distribution	100% outdoor (3km/h)
Average UEs per Sector	10
Number of Sectors	21
Number of Subbands (S)	12
RBs per Subband (C)	8

To assess the model's generalization performance, we compare the performance between the proposed VCSA and the "dedicated baseline" (DB). The DB involves training and testing for a single code size for each given scenario, e.g., UMa or UMi, exclusively. To ensure a comparable comparison, we maintain most parts in the NN architecture of the AE and adopt the same simulation configurations. It's worth noting that in the case of DB, the encoder input comprises CSI eigenvectors only, without the inclusion of the additional flag input. We adopt two code sizes in the study, namely 16 and 32, in the

TABLE II: Simulation Details.

Parameter	Value
AI/ML Model Type	CNN
Data Normalization	MinMax
Input/Output	Eigenvectors
Code Size	16, 32
Training Data Size	40093
Validation Data Size	7075
Testing Data Size	9112
Batch Size	256
Number of Epochs	300
Optimizer	Adamax
Loss Function	MSE
Quantization Type	Scalar Quantization (K-Means)
Quantization Bits	4, 5, 8
Payload size (bits)	80 (16 × 5bits), 128 (32 × 4bits), 256 (32 × 8bits)

two distinct AEs. The only difference is by modifying the size of the dense layer in the encoder part while the remaining NN architecture stays the same between the two AEs.

To evaluate the generalization performance across UMa and UMi scenarios, we conduct experiments by training two VCSAs, one with dataset from the target scenario only and another VCSA with combined dataset containing samples from both scenarios. For the VCSA with the target scenario (VCSAT), we train it with samples from a single scenario (UMa or UMi exclusively) and evaluate it on the test samples from the same scenario. In the case of the VCSA with combined scenarios (VCSAC), we train it using a mixed training dataset of UMa and UMi scenario, then evaluate its performance on each scenario's test samples separately. Additionally, we perform a naive transfer experiment by using the DB trained for one scenario to perform inference on test samples from the other scenario. For instance, we evaluate the inference performance using DB trained with samples from UMa scenario to performance inference on samples from UMi scenario.

B. Training Details

The UMa and UMi datasets undergo preprocessing (Eq. 2) to derive CSI eigenvectors. We augmented the dataset by duplicating the samples and used a "flag" to indicate the code size, i.e., 16 or 32 for each sample. To enable the encoder to generate an output with either code size, a mask that contains a set of 0's and 1's based on the value of the flag is applied to control the final size of the encoder output.

85% of the data (UMa and UMi) is used for training and the remaining 15% is designated for testing. 15% of the total training data is reserved for validation purposes as described in Table II. The overall training dataset contains the UMa and UMi samples, along with the corresponding flags.

The encoder input is represented as [CSI, flag]. The shape of the CSI part is $[32, 12, 2]$, where 32 denotes the number of transmitting antennas, 12 signifies the number of subbands, and 2 represents the real and imaginary components. The mask, which is based on the value of the flag influences the final encoder output. Mask 1, which corresponds to flag 0 contains sixteen 1's and sixteen 0's and mask 2, which corresponds to flag 1 contains thirty-two 1's. Thus, when mask 1 is applied, the final encoder output contains the first 16 codes in the original encoder output while the rest are zeros, and when mask 2 is applied, the final encoder output contains all 32 codes in the original encoder output.

TABLE III: Mapping of the CSI Payload Size

CSI Payload Size	Code Size	Quantization Bits per Code
80 bits	16	5
128 bits	32	4
256 bits	32	8

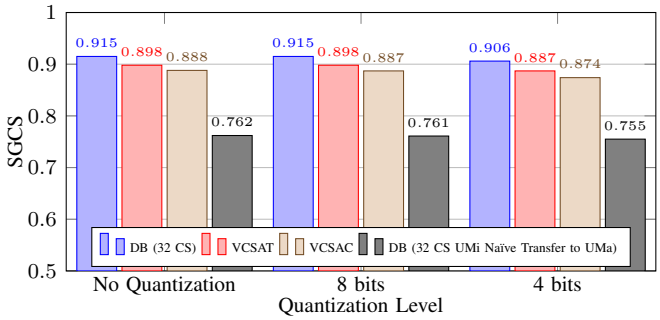

Fig. 3: Model Generalization Performance Comparison among VCSAT, VCSAC and Naïve transfer with code size = 32 (Target scenario = UMa).

The detailed VCSA architecture, including the encoder part and the decoder part, is illustrated in Fig. 2. It is important to note that during the training phase, the decoder always receives 32 codes as its inputs, which may comprise of either 16 codes in the original encoder output and 16 zeros or 32 codes.

Adamax optimizer is used in training the VCSA to dynamically adjust the learning rate, and Mean Square Error (MSE) is used as the loss function. Other training parameters are specified in Table II. Once the VCSA is trained, the encoder outputs for training samples from mask 2 are used to train the quantizer. This approach allows the quantizer to consider the codes generated using both mask 1 and mask 2.

For quantization, KMeans, an unsupervised learning technique, is employed to group the codes into K clusters based on Euclidean distance. The resulting K cluster centroids form a quantization dictionary. This quantization dictionary converts each code in the encoder output into an integer (from 0 to K-1) and each integer corresponds to its associated cluster number. This is referred as quantized encoder output. The quantized encoder output is then binarized to form a bit stream to be transmitted to the decoder side. The binarized bit stream received at the decoder side is first converted back to the corresponding integer values, then the same quantization dictionary is applied to convert the quantized results to floating-point values before feeding them into the decoder for the CSI reconstruction in the inference phase.

We adopt 4, 5, and 8 bits per code in combination with code sizes 16, and 32 to generate the final CSI payload sizes of 80 bits, 128 bits, 256 bits as shown in Table III.

C. Results

Fig. 3 depicts the SGCS comparison between DB for code size 32 and the introduced VCSA models (VCSAT and VC-

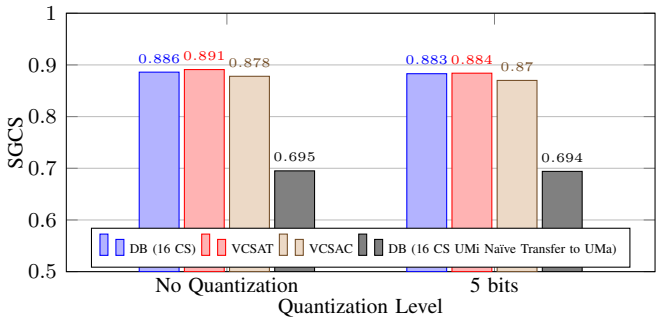

Fig. 4: Model Generalization Performance Comparison among VCSAT, VCSAC and Naïve transfer with code size = 16 (Target scenario = UMa).

SAC) that support both code sizes. In the figure, the blue bars depict the baseline SGCS performance from DB for 32 code size, indicated as DB (32 CS) in the legend. It can be noted that DB (32 CS) achieved the best performance attributed to the code size-specific training strategy. As discussed earlier, code size-specific model(s) will introduce additional overhead in storing multiple models. Thus, we also evaluated the performance between using the previously trained model for UMi scenario to performance inference on UMa scenario (denoted as Naïve transfer in Fig. 3). As samples from UMa scenarios were unseen during the training phase of the UMi model, the grey bars in Fig. 3 showed significant performance degradation compared to the baseline, DB (32 CS). We conducted similar performance comparison between DB for code size 16 and the proposed VCSA models (VCSAT and VCSAC) as depicted in Fig. 4. We noticed that VCSAT shows slightly higher SGCS compared to the baseline, DB (16 CS). This may due to the reason that training a unified NN supporting both code sizes allow some of the latent features for code size 32 being implicitly leveraged by code size 16 as well.

Overall, both VCSA models (VCSAT and VCSAC) achieve comparable performance as the DB models for code size 32 and 16 as depicted in Fig. 3 and Fig. 4. The details are outlined in Table IV. Compared to naïve transfer using DB for UMi (grey bars in Fig. 3 and Fig. 4), the VCSAC model (brown bars in Fig. 3 and Fig. 4), achieved an average of 22.7% SGCS gain. This gain emanates from the synergistic effects of training data stemming from both scenarios. It is also worth noting that the average absolute SGCS performance difference between the VCSAT and the VCSAC models is insignificant, ~ 0.012 across all payload sizes evaluated. It can also be noted from Fig. 3 and Fig. 4, in no quantization case, DB for code size 16 showed some performance degradation ($\sim 3\%$) compared to DB for code size 32. This is attributed to that using code size 16 introduces higher compression ratio, i.e., lower resolution of the original input compared to code size 32 which may impact the reconstruction accuracy. In contrast, the VCSA models (VCSAT and VCSAC) showed very insignificant performance difference between code size 16 and code size 32 ($\sim 1\%$).

Another observation is that both the DB and VCSAT models when employing quantization procedure, require separate

TABLE IV: Model Generalization Evaluation Results (UMa).

Payload Category	Payload Size	DB SGCS	VCSAC SGCS	Performance Difference (Gain %)
Small (Payload $\leq$ 80bits)	80bits	0.883	0.870	-0.013 (-1.472%)
Medium (100bits $\leq$ Payload $\leq$ 140bits)	128bits	0.906	0.874	-0.032 (-3.532%)
Large (Payload $\geq$ 230bits)	256bits	0.915	0.887	-0.028 (-3.060%)

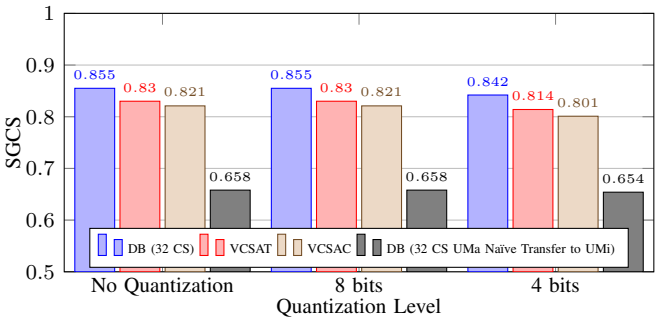

Fig. 5: Model Generalization Performance Comparison among VCSAT, VCSAC and Naïve transfer with code size = 32 (Target scenario = UMi).

quantization dictionaries for different code sizes while the VCSAC model efficiently utilizes a single quantization dictionary, which reduces the communication overhead associated with sharing the additional dictionary between the UE and gNB and storage requirements.

Likewise, we conduct the same experiment to evaluate the VCSAC performance for the UMi scenario, as depicted in Fig. 5 and Fig. 6. The observations align consistently with those from the UMa scenario, albeit with relatively lower SGCS values than the similar analyses for UMa scenario in general. An interesting observation arises from the examination of small payload sizes, as detailed in Table V that the VCSAC model outperforms the DB model by a marginal 0.377%. While this performance gain might appear slight, this can be attributed to several factors. The VCSAC model necessitates a doubled amount of training data from the same scenario to accommodate variable code sizes. This augmented training dataset enable the AE to characterize the input data better. A similar effect is applicable to the quantizer, which benefits from an increased number of samples. Consequently, the quantization loss can be further minimized, leading to these subtle performance gains. Our next research task is to compare the system level performance between the proposed approaches (i.e., VCSA and VCSAC) with traditional codebook-based approach.

Finally, we compare the neural network complexity in terms of the number of NN parameters, model storage estimation, and computational complexity using floating-point operations per second (FLOPs) between the proposed generalized model and the DBs, as depicted in Table VI. As the code size increases from 16 to 32, the encoder NN parameters, FLOPs, and storage requirements increase by 16.2%, 0.22%, and

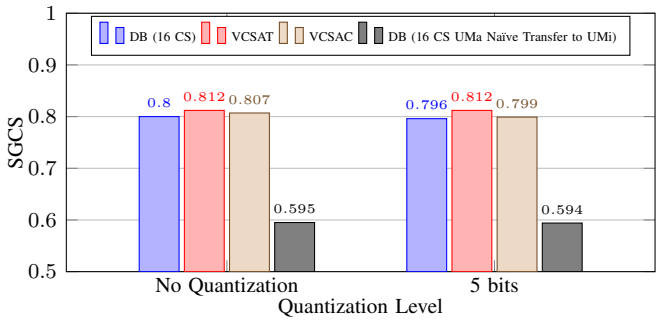

Fig. 6: Model Generalization Performance Comparison among VCSAT, VCSAC and Naïve transfer with code size = 16 (Target scenario = UMi).

TABLE V: Model Generalization Evaluation Results (UMi).

Payload Category	Payload Size	DB SGCS	VCSAC SGCS	Performance Difference (Gain %)
Small (Payload $\leq$ 80bits)	80bits	0.796	0.799	0.003 (0.377%)
Medium (100bits $\leq$ Payload $\leq$ 140bits)	128bits	0.842	0.801	−0.041 (−4.869%)
Large (Payload $\geq$ 230bits)	256bits	0.855	0.821	−0.034 (−3.977%)

16.1%, respectively. These increments are mainly due to the doubling the encoder output layer size. The NN complexity of the generalized encoder is the same as the 32-code size DB encoder, as the added flag input involves only a simple multiplication prior to the final encoder output. The computational complexity, measured in FLOPs, increases by 16, which is negligible. The generalized decoder maintains the same model complexity and computational complexity as the 32-code size DB decoder because the NN model itself remains unchanged.

IV. Conclusion

In this paper, we present a novel CNN-based autoencoder architecture for CSI compression and reconstruction, VCSA, which is a versatile encoder-decoder NN designed to adapt to different CSI encoder output sizes. In addition, to overcome the challenges of generalization across different deployment scenarios, we extend the original VCSA to VCSAC, which is trained using samples from multiple scenarios to enable an efficient CSI compression and reconstruction solution that can be applied in a range of deployment environments. Empirical results show that the VCSAC model can achieve performance levels comparable to code size and scenario-specific model baselines (DBs), which require training and storing multiple models. The results demonstrate potential benefits in utilizing advanced AI/ML-based approach, e.g., the VCSA, to reduce CSI feedback overhead and improve CSI reconstruction accuracy at the same time in 5G and beyond.

TABLE VI: Model Complexity and Storage Details

CSI Model	Number of NN Parameters	FLOPs	Storage (Mbytes)
16 code size DB encoder	2,431,248	354,489,616	9.3
16 code size DB decoder	3,762,562	580,166,400	14.3
32 code size DB encoder	2,824,480	355,276,064	10.8
32 code size DB decoder	4,155,778	580,952,832	15.8
VCSA (encoder)	2,824,480	355,276,080	10.8
VCSA (decoder)	4,155,778	580,952,832	15.8

References

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

[2] T. L. Marzetta and H. Yang, *Fundamentals of Massive MIMO*. Cambridge University Press, Feb. 2016.

[3] Z. Qin and H. Yin, "A review of codebooks for CSI Feedback in 5G new radio and beyond," *arXiv preprint arXiv:2302.09222*, Jun. 2023.

[4] P.-H. Kuo, H. Kung, and P.-A. Ting, "Compressive sensing based channel feedback protocols for spatially-correlated massive antenna arrays," in *Proc. IEEE Wirel. Commun. Netw. Conf. (WCNC)*, Jun. 2012, pp. 492–497.

[5] P. Liang, J. Fan, W. Shen, Z. Qin, and G. Y. Li, "Deep learning and compressive sensing-based CSI feedback in FDD massive MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 9217–9222, Aug. 2020.

[6] M. Soltani, V. Pourahmadi, A. Mirzaei, and H. Sheikhzadeh, "Deep learning-based channel estimation," *IEEE Commun. Lett.*, vol. 23, no. 4, pp. 652–655, Apr. 2019.

[7] J. Guo, C.-K. Wen, and S. Jin, "Deep learning-based CSI feedback for beamforming in single-and multi-cell massive MIMO systems," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 1872–1884, Jul. 2021.

[8] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wirel. Commun. Lett.*, vol. 7, no. 1, pp. 114–117, Feb. 2017.

[9] *New SI: Study on Artificial Intelligence (AI)/Machine Learning (ML) for NR Air Interface*. RP-213599, 3GPP, Qualcomm, RAN-94bis-e, Dec. 2021.

[10] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Deep convolutional autoencoder-based lossy image compression," in *Proc. IEEE Picture Coding Symp. (PCS)*, Sep. 2018, pp. 253–257.

[11] J. Guo, C.-K. Wen, S. Jin, and X. Li, "AI for CSI feedback enhancement in 5G-advanced," *IEEE Wirel. Commun.*, Dec. 2022.

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

[13] T. Wang, C.-K. Wen, S. Jin, and G. Y. Li, "Deep learning-based CSI feedback approach for time-varying massive MIMO channels," *IEEE Wirel. Commun. Lett.*, vol. 8, no. 2, pp. 416–419, Apr. 2018.

[14] S. Tang, J. Xia, L. Fan, X. Lei, W. Xu, and A. Nallanathan, "Dilated convolution based CSI feedback compression for massive MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 71, no. 10, pp. 11 216–11 221, Oct. 2022.

[15] 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)*, Jul. 2020, pp. 1–6.

[16] Z. Gao, L. Li, H. T. Wu, Xuezhen, and B. Hao, "A unified deep learning method for CSI feedback in massive MIMO systems," *ZTE Commun.*, vol. 20, no. 4, p. 110–115, Apr. 2022.

[17] M. Nerini, V. Rizzello, M. Joham, W. Utschick, and B. Clerckx, "Machine learning-based CSI feedback with variable length in FDD massive MIMO," *IEEE Trans. Wirel. Commun.*, May 2023.

[18] X. Liang, H. Chang, H. Li, X. Gu, and L. Zhang, "Changeable rate and novel quantization for CSI feedback based on deep learning," *IEEE Trans. Wirel. Commun.*, vol. 21, no. 12, pp. 10 100–10 114, Jun. 2022.

[19] V. Rizzello, M. Nerini, M. Joham, B. Clerckx, and W. Utschick, "User-driven adaptive CSI feedback with ordered vector quantization," *IEEE Wirel. Commun. Lett.*, Aug. 2023.

[20] *Study on Artificial Intelligence (AI)/Machine Learning (ML) for NR Air Interface*. R1-2205695, 3GPP, Ad-hoc Chair (CMCC), RAN-109bis-e, May 2022.

[21] *Study on Channel Model for Frequencies from 0.5 to 100 GHz (Release 16)*. 3GPP TR 38.901 V16.1.0, Tech. Rep., Nov. 2020.