RNN-Based Twin Channel Predictors for CSI Acquisition in UAV-Assisted 5G+ Networks

Muhammad Karam Shehzad, Luca Rose, Mohamad Assaad

RNN-Based Twin Channel Predictors for CSI Acquisition in UAV-Assisted 5G+ Networks

Muhammad K. Shehzad[1, 2], Luca Rose[1], and Mohamad Assaad[2]

[1]Nokia Bell Labs, France.
[2]Laboratory of Signals and Systems, CentraleSupelec, Gif-sur-Yvette, France.
Emails: muhammad.shehzad@nokia.com, luca.rose@nokia-bell-labs.com, mohamad.assaad@centralesupelec.fr

***Abstract*—Unmanned aerial vehicles (UAVs) evolution has gained an unabated interest for the use in several applications, such as agriculture, aerial surveillance, goods delivery, disaster recovery, intelligent transportation. The main features of this technology are high coverage, strong line-of-sight (LoS) links, promising throughput, cost-effective and flexible deployment. Currently, the Third Generation Partnership Project (3GPP) is working on the specification of release-17 (R-17) new radio (NR) for non-terrestrial networks (NTN). Therefore, owing to the drastic increase of UAV technology, in this paper, we propose channel state information (CSI) compression and its recovery with the aid of machine learning (ML)-based twin channel predictors. Due to the characteristic of gaining higher LoS communication paths in UAV network, the proposed strategy can bring potential benefits such as over-the-air (OTA)-overhead reduction, minimizing mean-squared-error (MSE) of a channel and maximizing precoding gain. Simulation-based results corroborate the validity of the proposed strategy, which can reap benefits in multiple factors.**

***Index Terms*—ML, CSI prediction, CSI compression, CSI reporting, MIMO, recurrent neural network (RNN), UAVs, 5G/6G.**

I. Introduction

A plethora of cellular users and their tremendous data rate requirements bring multiple challenges in the current cellular infrastructure. One of the promising solutions is to densely deploy small-cell base stations (SCBSs) – the idea is to bring the cellular users closer to the base station so that data rate can be maximized. However, such densification of SCBSs brings the challenge of their connectivity with the core network. Furthermore, in case of emergencies, e.g., tsunami, earthquake; cellular networks can largely be impacted. Therefore, unmanned aerial vehicles (UAVs) can provide a cost-effective solution to both of these issues by instant deployment [1], [2]. For instance, to deal with the connectivity of SCBSs with the core network, [3] introduced the use of UAVs as the communication hub between them. Similarly, [4] proposed an intelligent transportation system with the aid of UAVs.

Despite the fact that UAV can play a pivotal role in the modern era of wireless communication, there are multiple challenges associated with such a technology, and many of them have been addressed in the literature. For example, association (serving a group of network entities) of SCBSs with UAVs [5], placement of UAVs [6], [7], channel modeling [8], and so on. Nevertheless, to the best of our knowledge, channel state information (CSI) estimation, prediction, compression, and reporting have not been considered so far. In wireless communication, CSI is, by any measure, main element of every wireless communication technique, e.g., channel scheduling and precoding. To acquire the benefits of these wireless communication techniques, accurate CSI at the transmitter side is necessary. More specifically, currently, the Third Generation Partnership Project (3GPP) has also been working on the specification of release-17 (R-17) for non-terrestrial networks (NTN), which include satellite and high altitude platforms (HAPs). In such kind of NTN, CSI acquisition is of predominant importance.

The evolution of modern wireless communication techniques such as beamforming is beckoning the use of novel CSI reporting strategies. At present, in both communication techniques, i.e., time-division-duplex (TDD) and frequency-division-duplex (FDD), the base station sends a dedicated reference symbol (RS) to cellular users to get the estimate of the CSI. Nonetheless, to reduce over-the-air (OTA)-overhead cost, cellular users feedback a compressed version of estimated CSI. Notably, such compression can significantly reduce the precision of estimated CSI. Currently, 3GPP considers two types of CSI reports, i.e., type-I and type-II [9], [10]. Both types, however, consider strong compression of estimated CSI, which can result in deteriorating the performance of many communication techniques, e.g., precoding.

Motivated by the above issues and promising benefits of machine learning (ML) [11], [12], this paper focuses on employing ML in UAV-enabled wireless networks. In particular, this work tackles the issue of inaccurate CSI, which is mainly due to compression of estimated CSI. To this end, we address[1] the compression and recovery of CSI along with reducing the OTA-overhead cost. To acquire such an objective, we propose the use of twin[2] channel predictors at the UAV and the ground terminal (GT), i.e., SCBS or user equipment (UE). Here, importantly, in this study, we assume that such UAV can either belong to low altitude platforms or HAPs, which can hover from a few 100 meters to 20 kilometers. Feedback at the GT will be evaluated on the basis of the predicted channel.

[1]Within the context of ground cellular networks, the same idea has also been presented in [13], where the real and imaginary parts of channel model are trained/predicted separately using ML.

[2]They have same initialization conditions, i.e., weights, training data, etc.

As a toy example, if the predicted channel is good enough, then feedback can be eliminated; thus, bringing the necessary feedback-related overhead to zero. Otherwise, the proposed strategy can help in reducing the overhead as compared to the benchmark scheme.

The paper organization is as follows. In Section II, system model is discussed. The traditional scheme, which is used for CSI acquisition, is discussed in Section III. Recurrent neural network (RNN)-based proposed approach is discussed in Section IV. Simulation-based results are presented in Section V. Finally, conclusion is drawn in Section VI.

Notations: Matrices and vectors are denoted by boldface upper-case and boldface lower-case, respectively, and scalars with normal lower-case. Moreover, $\widehat{\mathbf{H}}$, $\widetilde{\mathbf{H}}$, $\mathbf{H}$, and $\overline{\mathbf{H}}$ depict the estimated, predicted, actual, and normalized channel, respectively. In addition, $Q_b(\cdot)$ and $Q_p(\cdot)$ are standard element-wise quantization functions followed in the benchmark and the proposed approach, respectively, where real and imaginary parts of the channel are quantized separately. Also, $[\cdot]^T$ denotes the transpose.

II. System Model

A. Communication Environment

Consider a UAV-assisted network where a multi-purpose[3] UAV is serving multiple ground network entities, e.g., SCBSs and UEs. In the rest of the paper, for the sake of simplicity, we will assume one ground network entity, remarking that this process would be followed by all the ground devices, served by the UAV. We will denote the ground network entity as GT in the rest of the paper. Without loss of generality, let us assume a multiple-input multiple-output (MIMO)-based communication configuration, which has N_T transmit and N_R receive antennas at the UAV and GT, respectively. Considering such an environment, a MIMO communication model can be written as

$$\mathbf{y}(n) = \mathbf{H}(n) \cdot \mathbf{p}(n) + \boldsymbol{\omega}(n) \tag{1}$$

where $\mathbf{y}(n) = \big[y_1(n), y_2(n), ..., y_{N_R}(n)\big]^T$ is the received signal at time instant n, having dimension $N_R \times 1$. Also, $\mathbf{H}(n) = \big[h_{n_r n_t}(n)\big]_{N_R \times N_T}$ depicts the channel, which has dimension $N_R \times N_T$, and $h_{n_r n_t} \in \mathbb{C}^{1\times 1}$ is the channel gain between the n_t^{th} transmit and n_r^{th} receive antenna. Further, $\mathbf{p}(n) = \big[p_1(n), p_2(n), ..., p_{N_T}(n)\big]^T$ and $\boldsymbol{\omega}(n)$ represent transmitted pilot symbols and additive white Gaussian noise, respectively.

B. Channel Model

Assume a UAV located at $(x_{\text{uav}}, y_{\text{uav}})$ hovering at an altitude h_{uav}, which is serving a GT located at (x, y). The horizontal distance between the UAV and a GT is $s = \sqrt{(x - x_{\text{uav}})^2 + (y - y_{\text{uav}})^2}$, and $\theta = \tan^{-1}\left(\frac{h_{\text{uav}}}{s}\right)$ represents the elevation angle (in degrees) between them. The air-to-ground channel model including path-loss and fast-fading effects is written as follows [14]

$$\mathbf{H} = \Big(\frac{\overline{\mathbf{H}}}{\sqrt{L_p}}\Big) \tag{2}$$

where $\overline{\mathbf{H}}$ and L_p are the normalized channel matrix and path-loss between the UAV and a GT, respectively. The line-of-sight (LoS) link between the UAV and a GT at an angle θ, is available by a certain probability, which is expressed as [15]

$$\varrho^{\text{LoS}} = \frac{1}{1 + \alpha \cdot \exp[-\beta(\theta - \alpha)]} \tag{3}$$

where α and β are the constants and their values are dependent on the environment (urban, suburban, rural, etc.). Besides, the probability of non-LoS (NLoS) is $\varrho^{\text{NLoS}} = 1 - \varrho^{\text{LoS}}$.

The path-loss for LoS and NLoS links are [16]:

$$L_p^{\text{LoS}}(\text{dB}) = 20\log_{10}\Big(\frac{4\pi f_c l}{c}\Big) + \eta^{\text{LoS}} \tag{4}$$

$$L_p^{\text{NLoS}}(\text{dB}) = 20\log_{10}\Big(\frac{4\pi f_c l}{c}\Big) + \eta^{\text{NLoS}} \tag{5}$$

where $l = \sqrt{s^2 + h_{\text{uav}}^2}$, f_c is the carrier frequency and c is the speed of light. Also, η^{LoS} and η^{NLoS} are the attenuation factors for the LoS and NLoS links, respectively, which are dependent on the environment. By using Equation (3), (4) and (5), the average path-loss is given as

$$L_p = \varrho^{\text{LoS}} \times L_p^{\text{LoS}} + \varrho^{\text{NLoS}} \times L_p^{\text{NLoS}}\,. \tag{6}$$

The fast-fading channel, $\overline{\mathbf{H}}$, is modeled as a Rician fading channel, which is composed of LoS and NLoS components, and is expressed as [14]

$$\overline{\mathbf{H}} = \sqrt{\frac{K}{1+K}} \cdot e^{j\phi} \cdot \overline{\mathbf{H}}^{\text{LoS}} + \sqrt{\frac{1}{1+K}} \cdot \overline{\mathbf{H}}^{\text{NLoS}} \tag{7}$$

where K is the Rician factor and is selected such that $\varrho^{\text{LoS}} = \frac{K}{1+K}$, ϕ is the phase shift of the signal between transmit and receive antenna. Additionally, $\overline{\mathbf{H}}^{\text{LoS}}$ is the constant term corresponding to LoS component, and $\overline{\mathbf{H}}^{\text{NLoS}}$ is the Rayleigh fading channel representing the multi-path reflections for NLoS components.

Assuming a FDD system, owing to the absence of channel reciprocity, UAV sends RS to the GT for channel acquisition. Consequently, the GT estimates the channel using a dedicated channel estimator and feedback the compressed version of the estimated channel to reduce the OTA-overhead. In the following, we briefly summarize the CSI acquisition strategy, followed in the benchmark scheme. Later on, we will address the CSI compression and its recovery in the proposed approach.

[3]The UAV can act as a relay between SCBSs and the ground core-network to route the uplink/downlink traffic of cellular users [3]. Also, it can be used as a temporary aerial base station (ABS) in the scenario, where ground base station is somehow damaged [6], etc. Moreover, a UAV can be playing the role of traffic monitoring for intelligent transportation system [4], etc.

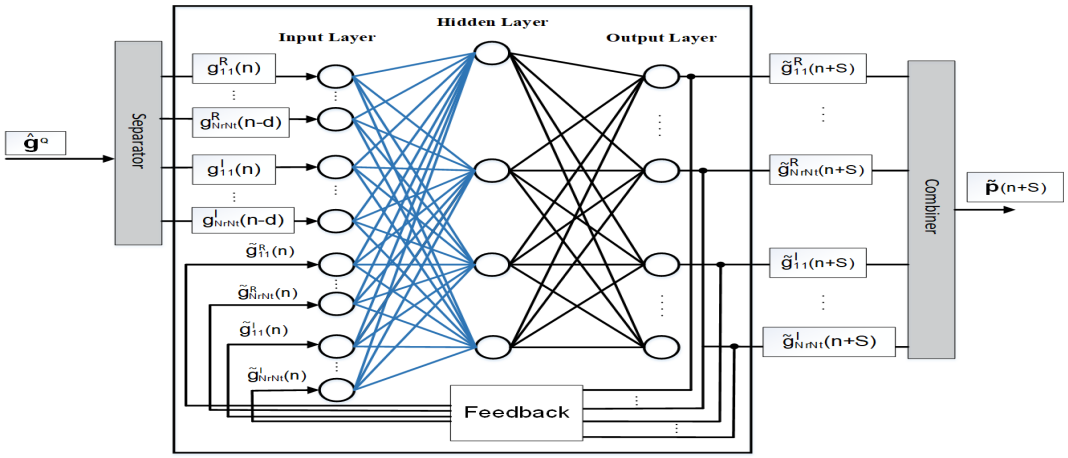

Fig. 1. A graphical representation of an RNN-based channel predictor.

III. Benchmark Scheme for CSI Acquisition

Traditionally, to the best of our knowledge, CSI acquisition has not been considered in UAV communication. All the same, for the sake of comparison, we give an overview of the benchmark scheme, which can be followed for the CSI compression in UAV communication. However, importantly, such benchmark scheme has been considered in many other contexts, e.g., type-I and type-II CSI feedback [9], [10], but not in the domain of UAV communication. Briefly, in the benchmark scheme, the UAV transmits pilot symbols to the GT to estimate the channel. In consequence, the GT estimates the channel, denoted by $\widehat{\mathbf{H}}_{\text{GT}}$, using a dedicated channel estimator, e.g., Kalman filter (KF). Also, to reduce the OTA-overhead, the estimated channel is compressed using a quantization function and is written as

$$\widehat{\mathbf{H}}^{Q}(n) = Q_b(\widehat{\mathbf{H}}_{\text{GT}}(n)) \tag{8}$$

where $Q_b(\cdot)$ represents the quantization function. Subsequently, this quantized version is fed back to the UAV, which is considered as an estimated channel at the UAV. Hence, the acquired channel at the UAV can be written as

$$\widehat{\mathbf{H}}_{\text{uav}}(n) = Q_b(\widehat{\mathbf{H}}_{\text{GT}}(n))\,. \tag{9}$$

However, such compression can deteriorate the performance of the estimated channel. Further, in massive MIMO, feedback grows largely; thereby, such compression effects the performance of MIMO precoder. Moreover, in excessive feedback demand, a portion of UAV's energy can decrease; thus, UAV cannot stay aloft for a longer time period. For instance, when the performance of MIMO precoder is poor, to achieve a given bit error rate, UAV would need more transmit power; consequently, UAV's flying time would decrease. Motivated by these issues, in the following, we present our proposed approach, which is based on twin channel predictors.

IV. RNN-Based CSI Compression and Recovery

In the beginning of the proposed approach, we assume that both the network entities, i.e., UAV and the GT, are operating on traditional CSI acquisition scheme (as discussed in Section III). Consequently, UAV and GT can establish a past amount of channel realizations; thus, we assume that the compressed version of the estimated channel, i.e., $\widehat{\mathbf{H}}^{Q}$, is available on both sides. Additionally, we consider a twin channel predictor, which is deployed on UAV and the GT. The feedback at the GT will be evaluated based on the predicted channel at GT. As a toy example, if the prediction at the GT is perfect, then there is no need to feedback anything. On the other hand, if there is anything to feedback, then the update function (the difference between predicted channel and estimated channel) will introduce less noise in the compression as compared to (8). Below, we first describe the channel predictor, which is used in this study. Later on, compression strategy and its recovery will be explained.

A. RNN-Based Channel Prediction

Recently, ML has emerged as a paradigm shift in the modern era of wireless communication [5], [11], [12]. Although ML has been considered in many aspects of wireless communication, applications at the physical layer are still lagging. Therefore, this work introduces a channel prediction-aided CSI reporting, which is shown to reduce both feedback overhead and improve CSI precision. Specifically, within the domain of ML, RNN has proven to be a powerful technique for making the time-series prediction [17]. As RNN have the characteristic of making the prediction based on the current input as well as previous computation; hence, RNN is different from the traditional neural network (TNN), i.e., feedforward neural network. In the TNN, every input of the neural network is independent of each other. In many cases, upcoming input can be dependent on the previous one (for example, making the prediction of the next word in the sentence).

The objective of the RNN is to give multi-step $(n+S)$ ahead prediction of the channel, represented by $\widetilde{\mathbf{H}}(n+S)$, which is as precise as possible to the actual value, i.e., $\widehat{\mathbf{H}}^{Q}(n+S)$. The pictorial representation of a RNN is given in Fig. 1. In

the domain of ML, such an architecture is known as many-to-many RNN as it is consisted of multiple-inputs and multiple-outputs. As depicted in Fig. 1, RNN is consisted of an input layer, which has L input neurons, a hidden layer consisted of M hidden layer neurons, and an output layer, comprises of O output neurons. Further, input neurons, L, are consisted of external inputs and the internal feedback. On the other hand, to train and test the RNN, $\widehat{\mathbf{H}}^{Q}(n)$, which can be represented as

$$\widehat{\mathbf{H}}^{Q}(n) = \begin{bmatrix} \widehat{h}^{Q}_{11}(n) & \cdots & \widehat{h}^{Q}_{1N_T}(n) \\ \widehat{h}^{Q}_{21}(n) & \cdots & \widehat{h}^{Q}_{2N_T}(n) \\ \vdots & \ddots & \vdots \\ \widehat{h}^{Q}_{N_{R}1}(n) & \cdots & \widehat{h}^{Q}_{N_R N_T}(n) \end{bmatrix} \tag{10}$$

where $\widehat{h}^{Q}_{n_r n_t}(n) \in \mathbb{C}^{1\times 1}$ denotes the compressed version of estimated channel for transmit antenna n_t and receive antenna n_r, is taken along with its $d-$step delayed channel realizations, i.e., $\widehat{\mathbf{H}}^{Q}(n-1), \widehat{\mathbf{H}}^{Q}(n-2), ..., \widehat{\mathbf{H}}^{Q}(n-d)$. However, as ML cannot deal with matrices, therefore, the combined data, i.e., $\widehat{\mathbf{H}}^{Q} = \{\widehat{\mathbf{H}}^{Q}(n), \widehat{\mathbf{H}}^{Q}(n-1), \widehat{\mathbf{H}}^{Q}(n-2), ..., \widehat{\mathbf{H}}^{Q}(n-d)\}$, is unrolled into a vector as

$$\widehat{\boldsymbol{g}}^{Q} = \text{unroll}(\widehat{\mathbf{H}}^{Q}) = [\widehat{h}^{Q}_{11} \quad \widehat{h}^{Q}_{12} \quad ... \quad \widehat{h}^{Q}_{N_R N_T}] \,. \tag{11}$$

Importantly, the training and prediction of a complex-valued RNN is not well implemented in the current ML-based software tools, therefore, in our work, we use real-valued based RNN. Thus, to this end, we separate the complex-valued vector $\widehat{\boldsymbol{g}}^{Q}$ into real and imaginary parts, using a dedicated *separator* function (as drawn in the left-most part of Fig. 1). Therein, the resultant vector can be written as

$$\boldsymbol{g} = [g^{R}_{11}(n) \quad ... \quad g^{R}_{N_R N_T}(n-d), g^{I}_{11}(n) \quad ... \quad g^{I}_{N_R N_T}(n-d)] \tag{12}$$

where $g^{R}_{n_r,n_t}$ and $g^{I}_{n_r,n_t}$ represent the real and imaginary parts of a channel $\widehat{h}^{Q}_{n_r n_t} \in \mathbb{C}^{1\times 1}$. In addition, the feedback from the output, can be written in vector form as

$$\widetilde{\boldsymbol{g}}(n) = [\widetilde{g}^{R}_{11}(n) \quad ... \quad \widetilde{g}^{R}_{N_R N_T}(n), \widetilde{g}^{I}_{11}(n) \quad ... \quad \widetilde{g}^{I}_{N_R N_T}(n)] \tag{13}$$

where $\widetilde{g}^{R}_{n_r n_t}(n)$ and $\widetilde{g}^{I}_{n_r n_t}(n)$ depict the real and imaginary parts of the predicted channel ($\widetilde{h}_{n_r n_t}(n) \in \mathbb{C}^{1\times 1}$) for the n_t^{th} and n_r^{th} transmit and receive antenna, respectively. Finally, together with the external input, the combined input, i.e.,

$$\mathbf{i}(n) = [\boldsymbol{g}, \widetilde{\boldsymbol{g}}(n)] \tag{14}$$

is fed as an input to the neural network. Consequently, multi-step ahead prediction of the real and imaginary parts of a channel, which are denoted by $\widetilde{g}^{R}_{n_r n_t}(n+S)$ and $\widetilde{g}^{I}_{n_r n_t}(n+S)$, respectively, can be obtained. Afterwards, real and imaginary parts of the channel are combined using a function, we call it *combiner*, as portrayed on the right-most part of Fig. 1. By doing some processing, S-step ahead predicted channel can be written in the vectorized form as $\widetilde{\mathbf{p}}(n+S)$, which can be seen at the output of *combiner*. Thereby, multi-step ahead predicted channel, for a MIMO configuration, can be expressed (by converting the $\widetilde{\mathbf{p}}(n+S)$) into matrix form as $\widetilde{\mathbf{H}}(n+S)$.

On the other side, prediction of any ML-based predictor is totally dependent on the weights of input and hidden layers, and the activation function. A weight value is assigned for a connection between the output of a neuron in the predecessor layer and input of a neuron in the successor layer. For instance, we denote the weight of l^{th} input and m^{th} hidden neuron as w_{ml}, and w_{om} represents the weight for the connection between o^{th} output neuron and m^{th} hidden layer neuron, where $1 \leq l \leq L$, $1 \leq m \leq M$, and $1 \leq o \leq O$. Also, for the hidden layer neuron, we use rectified linear unit as an activation function, which is written as

$$a(z) = max(0, z) \tag{15}$$

where

$$z = \mathbf{w}_m \cdot \mathbf{i}(n) \tag{16}$$

where $\mathbf{w}_m = [w_{m1}, ..., w_{mM}]$ is the weight vector for m^{th} hidden layer neuron. Thus, the output for the m^{th} hidden neuron, at n^{th} time instant, can be obtained as follows

$$\xi_m(n) = a(\mathbf{w}_m \cdot \mathbf{i}(n)) \,. \tag{17}$$

Subsequently, passing the above activation function to the o^{th} output neuron, $n+S$ time ahead predicted value can be obtained as

$$J^{o}(n+S) = \sum_{m=1}^{M} w_{om} \cdot \xi_m(n) \,. \tag{18}$$

Lastly, by substituting (17) into (18), S-step ahead prediction at an output neuron, representing the prediction for the real or imaginary part of the channel, can be written as

$$\widetilde{g}^{o}(n+S) = \sum_{m=1}^{M} w_{om} \cdot a(\mathbf{w}_m \cdot \mathbf{i}(n)) \,. \tag{19}$$

The functionality of a RNN-based MIMO channel predictor is composed of two stages, i.e., training and prediction. In the beginning, once the hyperparameters of RNN, e.g., number of hidden layers and neurons, have been selected, then the training stage can begin. In the training stage, input dataset, i.e., channel values, which are consisted of training and validation (optional) set, are fed as an external input to RNN, along with actual labels. As an outcome, RNN processes each example (or a batch of examples, which depends on the batch size), and compares the resulting predicted value with the actual label. Next, the error between the prediction and corresponding label is backpropagated (feedback) to the network such that the weights of each connection can be updated in an iterative manner. This process is repeated until mean-squared-error (MSE) between the predicted and actual label, is minimized. Lastly, once an RNN has been trained, then it can be used as a predictor for test (or unknown) inputs. The resultant RNN[4] can give a multi-step ahead prediction of the channel.

[4]To know more about training and prediction of an RNN, interested readers can refer to, e.g., [17].

Having the trained channel predictor at the UAV and GT, in the following, we first explain the compression of CSI at the GT.

B. CSI Compression at GT

The twin trained RNN-based channel predictors on both sides of the communication system, predict the next channel realization, e.g., at time n. Clearly, since the channel predictors are twin and they are using the same input for the prediction, therefore, predicted channel on both sides would be same, i.e., $\widetilde{\mathbf{H}}_{\text{uav}}(n) = \widetilde{\mathbf{H}}_{GT}(n)$, where $\widetilde{\mathbf{H}}_{\text{uav}}(n)$ and $\widetilde{\mathbf{H}}_{GT}(n)$ represent the predicted channel at the UAV and GT, respectively, at time n. In addition, at time n, UAV sends the dedicated RS to the GT for CSI acquisition. Accordingly, GT estimates the channel at time n using the dedicated channel estimator, i.e., KF. Later on, an update function, at the GT, is computed as

$$\mathbf{\Gamma}(n) = \zeta\big[\widetilde{\mathbf{H}}_{GT}(n), \widehat{\mathbf{H}}_{GT}(n)\big] \tag{20}$$

where $\zeta[\cdot]$ is a function, which represents the measure of the distance between the two channel realizations, e.g., a difference. Hence, (20) can be written as

$$\Delta\mathbf{C}(n) = \widetilde{\mathbf{H}}_{GT}(n) - \widehat{\mathbf{H}}_{GT}(n)\,. \tag{21}$$

Then, the update, given in (21), is compressed at the GT as

$$\mathbf{U}_c(n) = Q_p(\widetilde{\mathbf{H}}_{GT}(n) - \widehat{\mathbf{H}}_{GT}(n)) \tag{22}$$

where $Q_p(\cdot)$ represents the quantization function. The benefits of using the above compression strategy as compared to (8) are twofold. Firstly, quantization error, which will be introduced in (22), can be lower than (8). This is because of the fact that in (22), the amplitude of the entries can be very small; therefore, less quantization error will occur. Secondly, having smaller values in (22) will require fewer bits to quantize; thereby, the overhead required for the CSI compression can be reduced significantly. Moreover, if the prediction and estimation at the GT are perfect, i.e., $\widetilde{\mathbf{H}}_{GT}(n) = \widehat{\mathbf{H}}_{GT}(n)$, then there is no need to feedback anything; therein, feedback-related overhead can be eliminated. This scenario, in particular, can happen in the network, where GTs are static. For instance, as presented in [3], [5]–[7], where ground SCBSs are static, and UAVs are used as communication hubs to route the uplink/downlink traffic of cellular users between the ground core-network and the SCBSs. Or, alternatively, when GTs are static cellular users (e.g., cellular users watching a football match in the stadium, attending a seminar, or cellular users sitting at home due to the lockdown of global coronavirus pandemic). This is due to the reason that for static GTs, variation in the channel can be very small; therefore, prediction can be more accurate. Thus, overhead can be reduced tremendously. Below, we present the recovery of quantized updates (i.e., (22)) at the UAV.

C. CSI Recovery at UAV

The recovery of the quantized update, i.e., (22), at the UAV, is divided into two cases: when the UAV receives the feedback from GT and when it does not.

1) Case-1: In the first case, when the UAV receives the feedback from the GT, the estimated channel at the UAV, at time n, can be obtained through the following equation

$$\widehat{\mathbf{H}}^{p}_{\text{uav}}(n) = \widetilde{\mathbf{H}}_{\text{uav}}(n) - \mathbf{U}_c(n)\,. \tag{23}$$

By substituting (22) into (23), the estimated channel at the UAV, can be expressed as

$$\widehat{\mathbf{H}}^{p}_{\text{uav}}(n) = \widetilde{\mathbf{H}}_{\text{uav}}(n) - Q_p(\widetilde{\mathbf{H}}_{GT}(n) - \widehat{\mathbf{H}}_{GT}(n))\,. \tag{24}$$

Based on the above equation, we make two remarks. Firstly, the better the prediction at the GT less would be the error in the update. Therefore, fewer bits will be required to quantize the update. Secondly, if the quantizer is using an infinite amount of bits for the quantization, then (22) would not give any advantage. In simple words, quantization in the benchmark scheme and the proposed scheme would be the same, i.e., $Q_b(x) = Q_p(x) = x$, where x is the data to be quantized.

2) Case-2: In the second case, when UAV does not receive any feedback from the GT, which can be in the scenario when $\widetilde{\mathbf{H}}_{GT}(n) = \widehat{\mathbf{H}}_{GT}(n)$; thus, in such case, estimated channel at the UAV would be simply

$$\widehat{\mathbf{H}}^{p}_{\text{uav}}(n) = \widetilde{\mathbf{H}}_{\text{uav}}(n)\,. \tag{25}$$

Therefore, in the second case, feedback-related overhead has been eliminated. As the UAVs provide strong LoS communication paths, also, in static GTs (e.g., when UAVs serve SCBSs), there can be negligible variance in the channel; thereby, it is possible that the predicted channel could be much closer to the estimated channel. Thus, UAV can save energy; consequently, UAV can stay aloft for a prolonged time.

V. Results And Analysis

In this section, the performance of the proposed approach is compared with the benchmark scheme. We assume a static UAV hovering at 150 meters, a GT randomly moving in square of length 500 meters, and $\overline{\mathbf{H}}^{\text{LoS}}$ equivalent to identity matrix. The rest of the simulation parameters are given in Table I. Importantly, for the sake of observing compression errors, we consider perfect channel estimation. Moreover, to analyze the results, we consider MSE (ξ_{mse}) as evaluation parameter. The MSE can be obtained by using the formula given below

$$\xi_{\text{mse}} = \frac{1}{N}\sum_{n=1}^{N}\Big\|\mathbf{H}(n) - \widehat{\mathbf{H}}^{'}_{\text{uav}}(n)\Big\|^2 \tag{26}$$

where, in the case of benchmark scheme, $\widehat{\mathbf{H}}^{'}_{\text{uav}}(n) = \widehat{\mathbf{H}}_{\text{uav}}(n)$, while for the proposed approach, $\widehat{\mathbf{H}}^{'}_{\text{uav}}(n) = \widehat{\mathbf{H}}^{p}_{\text{uav}}(n)$. Besides, in our experiments, we observed a prediction error of 2.3×10^{-3} in RNN. Nevertheless, designing an accurate channel predictor is not the goal of this study. Therefore, below, we evaluate the performance by using ξ_{mse}.

Fig. 2 shows ξ_{mse} when quantization bits, denoted by Q, are increased. The decreasing trend in the ξ_{mse} can be observed when Q is increasing. Nonetheless, the proposed approach is outperforming the benchmark. For instance, when $Q = 1$ then the benchmark scheme has $\xi_{\text{mse}} \approx 0.11$ while the proposed

TABLE I
SIMULATION PARAMETERS.

Parameter	Value	Description
$d, \{x_{\text{uav}}, y_{\text{uav}}\}, D_{\text{Size}}$	$1, \{230, 340\}, 5000$	Tapped delay, position of UAV, and size of data-set
$\{N_T, N_R, \phi, c, h_{\text{uav}}\}$	$\{2, 2, \pi/4, 3 \times 10^8 \text{ m/s}, 150 \text{ m}\}$	Description given in Section II
$\{f_c, \alpha, \beta, \eta^{\text{LoS}}, \eta^{\text{NLoS}}\}$	$\{2 \text{ GHz}, 9.61, 0.16, 1 \text{ dB}, 20 \text{ dB}\}$ [15]	Description given in Section II
RNN architecture	3-layer	RNN's input, hidden, and output layer
Optimization algorithm, N_e	Adaptive moment estimation (*Adam*) [18], 15	Algorithm to update weights of RNN, and number of epochs
$L, M, \&O$	24, 24, 8	Number of neurons in input, hidden and output layer, respectively
$D_{\text{train}}, D_{\text{valid}}, D_{\text{test}}$	$80\%, 10\%, 10\%$	Distribution of data-set: training, validation, and test, respectively
B, ML Platform	12, *TensorFlow* with *Python*	Batch size, and implementation platform for RNN algorithm

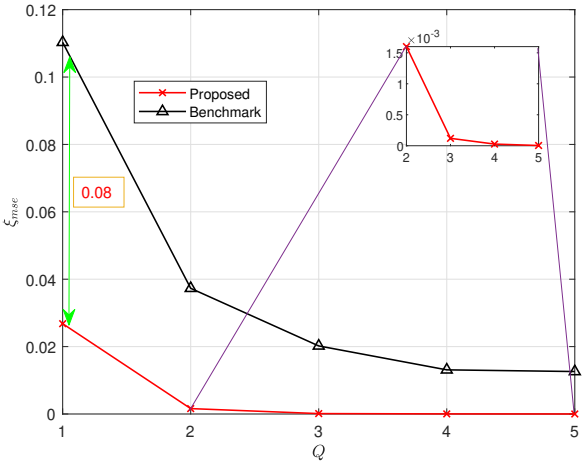

Fig. 2. Performance evaluation of benchmark versus proposed approach using ξ_{mse} (given in (26)) when quantization bits Q are varied.

approach has $\xi_{\text{mse}} \approx 0.03$, and this reduces further when Q increases. Thus, it can be concluded that the benchmark scheme demands more overhead. In summary, the proposed approach has not only saved the overhead bits, Q, but also recovered the actual channel 100% by using only 2 bits; thus, reaping benefit of precoding gain.

VI. CONCLUSION

Seeing the growing interest in UAV communication and the predominant importance of CSI acquisition, this work addressed the CSI compression and its recovery with the aid of ML-based twin channel predictors. Simulation-based results validated (100% recovery of the compressed channel is observed by using fewer overhead bits) the use of the proposed approach, which can bring potential benefits in the UAV scenarios where GTs are static or even moving at a predictable speed.

REFERENCES

[1] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE communications surveys & tutorials*, vol. 21, no. 3, pp. 2334–2360, 2019.

[2] M. K. Shehzad, A. Ahmad, S. A. Hassan, and H. Jung, "Backhaul-aware intelligent positioning of UAVs and association of terrestrial base stations for fronthaul connectivity," *IEEE Transactions on Network Science and Engineering*, pp. 1–14, 2021.

[3] M. Alzenad, M. Z. Shakir, H. Yanikomeroglu, and M.-S. Alouini, "FSO-based vertical backhaul/fronthaul framework for 5G+ wireless networks," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 218–224, 2018.

[4] M. A. Cheema, M. K. Shehzad, H. K. Qureshi, S. A. Hassan, and H. Jung, "A drone-aided blockchain-based smart vehicular network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4160–4170, 2021.

[5] M. K. Shehzad, S. A. Hassan, A. Mahmood, and M. Gidlund, "On the association of small cell base stations with UAVs using unsupervised learning," in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*. IEEE, 2019, pp. 1–5.

[6] M. K. Shehzad, S. A. Hassan, M. Luque-Nieto, J. Poncela, and H. Jung, "Energy efficient placement of UAVs in wireless backhaul networks," in *Proceedings of the 2nd ACM MobiCom Workshop on Drone Assisted Wireless Communications for 5G and Beyond*, 2020, pp. 1–6.

[7] M. K. Shehzad, M. W. Akhtar, and S. A. Hassan, *Performance of mmWave UAV-assisted 5G hybrid heterogeneous networks*. John Wiley & Sons, Ltd, 2021, ch. 6, pp. 97–118. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119751717.ch6

[8] M. T. Dabiri, H. Safi, S. Parsaeefard, and W. Saad, "Analytical channel models for millimeter wave UAV networks under hovering fluctuations," *IEEE Transactions on Wireless Communications*, vol. 19, no. 4, pp. 2868–2883, 2020.

[9] " 5G/NR - CSI RS Codebook ," https://www.sharetechnote.com/html/5G/5G_CSI_RS_Codebook.html, accessed on 2020-09-22. [online].

[10] https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3216, accessed on 2020-09-22. [online].

[11] L. Rose, S. Lasaulce, S. M. Perlaza, and M. Debbah, "Learning equilibria with partial information in decentralized wireless networks," *IEEE Communications Magazine*, vol. 49, no. 8, pp. 136–142, 2011.

[12] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[13] M. K. Shehzad, L. Rose, and M. Assaad, "Dealing with CSI compression to reduce losses and overhead: An artificial intelligence approach," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2021, pp. 1–6.

[14] L. Sboui, H. Ghazzai, Z. Rezki, and M. Alouini, "Achievable rates of UAV-relayed cooperative cognitive radio MIMO systems," *IEEE Access*, vol. 5, pp. 5190–5204, 2017.

[15] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal LAP altitude for maximum coverage," *IEEE Wireless Communications Letters*, vol. 3, no. 6, pp. 569–572, 2014.

[16] A. Al-Hourani, S. Kandeepan, and A. Jamalipour, "Modeling air-to-ground path loss for low altitude platforms in urban environments," in *2014 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2014, pp. 2898–2904.

[17] J. T. Connor, R. D. Martin, and L. E. Atlas, "Recurrent neural networks and robust time series prediction," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 240–254, 1994.

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv 2017, arXiv:1412.6980.