IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 30, 2021, accepted June 4, 2021, date of publication June 8, 2021, date of current version June 16, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3087630

A Machine Learning Method to Synthesize Channel State Information Data in Millimeter Wave Networks

UMAIR F. SIDDIQI[1], (Member, IEEE), SADIQ M. SAIT[1,2], (Senior Member, IEEE), AND KHALED ABDUL-AZIZ AL-UTAIBI[3], (Member, IEEE)

[1]Center for Communications and IT Research, Research Institute, King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia
[2]Department of Computer Engineering, King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia
[3]Department of Computer Science and Software Engineering, University of Ha'il, Ha'il 81451, Saudi Arabia

Corresponding author: Sadiq M. Sait (sadiq@kfupm.edu.sa)

This work was supported by the Deanship of Scientific Research, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, under Project SB191038.

ABSTRACT In millimeter-wave (MMW) networks, the channel state information (CSI) carries essential information from the user to the base station (BS). The CSI values depend highly on the geometrical and physical features of the environment. Therefore, it is impossible to generate CSI data for computer simulations or analysis through mathematical models. The CSI in MMW networks can only be acquired through physical measurement(s) or with the help of expensive and complicated ray-tracing software. For many users, both these options are infeasible. This work aims to propose a simple and fast method that can generate artificial samples from the real data samples while ensuring that the artificial samples look similar to the real ones. The proposed method helps increase the size of existing CSI datasets and likely to benefit the evolution of deep learning models that need a large amount of training/testing data. The proposed method comprises two parts. (i) The first part applies data clustering and transformations such as principal component analysis (PCA)-based dimensionality reduction and probability integral transform (PIT) to convert the real data into a multivariate normal distribution of a smaller number of variables, and (ii) The second part synthesizes artificial data by learning from the multivariate normal distribution of the first part. The last step in the second part is to apply PIT and inverse PCA transformations to transform the artificial data into the same space as the input data. We compared the proposed method's performance with the well-known Kernel density estimation (KDE)-based methods that use Scott's rule and Silverman's rule to choose the bandwidth parameter value. The results show that the artificial samples generated by the proposed method exhibit very high similarity with the real ones as compared to the KDE-based methods.

INDEX TERMS Millimeter wave networks, wireless communications, machine learning, principal component analysis, artificial data.

I. INTRODUCTION

Millimeter-wave (mmWave) and massive-multiple-input multiple-output (MIMO) are two critical components of fifth-generation (5G) networks. The spectrum of mmWaves lies between 30 GHz and 300 GHz [1] and meets the wider bandwidth and low latency requirements of the 5G communication systems. However, the implementation of mmWave communications also has technical issues, such as (i) mmWaves suffer very high path loss as compared to the traditional wireless communications that operate below 6 GHz and (ii) are sensitive to blockages because of its lesser penetration capability [2]. A technique known as beamforming (BM) can improve the range of the mmWave by concentrating the transmission power within a narrow beam towards the receiver. In traditional MIMO networks, the interference plays a major role in the signal-to-interference-noise-ratio (SINR) values. However, in the mmWave-based MIMO systems, BM limits the role of interference, and SINR values depend on the signal loss and blockages. The mmWave is adopted in two communication standards: (a) IEEE 802.15.3-2016; and (b) IEEE 802.11ay. The IEEE 802.15.3-2016 facilitates data and multimedia transmission using the 2.4 GHz and 60 GHz frequencies in wireless

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Feng.

personal area networks (WPAN). The IEEE 802.11ay is a wireless local area network (WLAN) standard that has a frequency of around 60 GHz and bandwidth of 2.16 GHz. The massive-MIMO uses a large number of antennas at the small-base-station (SBS), and improves the performance of mmWave networks by suppressing interference in dense networks, providing wireless back-haul, and improving energy efficiency.

Massive-MIMO is another key technology of 5G networks in which the small-base-stations (SBSs) employ an array of antennas. In massive-MIMO systems, each user-equipment (UE) receives signals from multiple antennas of each SBS. The UE computes the channel state information (CSI) using data-packets from each antenna of different SBSs. The UE should transmit the CSI back to the SBSs through feedback to enable the SBSs to perform tasks vital to maintain performance in the network, e.g., BM, spatial multiplexing, and user-association.

In massive-MIMO networks, researchers developed the geometry-based stochastic channel model (GSCM) that contains the physical models of the components of the network [3]. However, in the realm of mmWave, the CSI is highly dependent on the geometrical and physical features of the environment, limiting the usefulness of the mathematical models. Therefore, researchers have recommended collecting the CSI through real-world measurements or ray-tracing software. Both these approaches require expensive software tools, signal measurement instruments, and human-efforts [4]. In short, the artificial CSI data of mmWave cannot be generated using any analytical model or mathematical equations. It is impossible to extend a dataset without recapturing data through physical experimentation or using commercial ray-tracing software. Both these methods are inaccessible for most of the users, and they also require expertise in wireless communications.

In the field of data science, methods for the generation of artificial data comprise two steps. The first step is to estimate the probability-density-function (PDF) from the given data, and the second is to use the estimated PDF to sample new data points [5]. The density estimation can either be explicit or implicit [6]. Explicit methods of density estimation are of two types: (i) parametric and,(ii) non-parametric. Parametric methods postulate a model for the data and then use the data to fit the model's parameters. The PDF of the data is usually unknown, making it very difficult to estimate the it by parameter-fitting of known function types. The non-parametric methods estimate the PDF by finding a model that fits the distribution of the given data. Histograms are the most basic type of non-parametric method of density estimation. The kernel density estimation (KDE) method is a popular non-parametric method that assumes that we can express the data's distribution as a sum of each data point's kernel function, and the most common kernel function used is the normal distribution function. It has a critical parameter bandwidth that controls the smoothness of the density estimate. When the bandwidth is too small, the density estimate becomes over-fitted and has many peaks. When the bandwidth is too large, then it skips critical details. For high-dimensional and large-size datasets, it is difficult and computationally intensive to use the KDE method [7]. The generative adversarial Network (GAN) is an example of methods that implicitly estimate the PDF of the given data. The GANs consist of two models: Generator and Discriminator. The role of the generator is to synthesize artificial data samples, and that of the discriminator is to distinguish the artificial data samples from the real ones. The generator and discriminator compete against each other until the system reaches an equilibrium state. Although the idea of GANs is impressive, they have some critical problems [8]–[10], such as the following: (i) Mode collapse, which is the lack of diversity in the artificial data, and this happens because the generative adversarial networks (GANs) generate only a few types of signals; (ii) The architecture of the discriminator and generator should be competitive to each other, and imbalance architectures lead to unstable training; (iii) The GANs also consume a lot of training time to be able to generate high-quality artificial data; (iv) They have complex architectures that require too many design choices and optimization of many parameters.

This work proposes a simple and fast method to generate high-quality artificial CSI data of mmWave networks. The proposed method is motivated by the GAN's architecture and does not estimate the PDF of the input data. It is also fast and simple, like the statistical methods. The proposed method consists of two parts. The first part processes the real data, and the second generates the artificial data. The first part performs three tasks: (i) Division of the data into clusters; (ii) Dimensionality reduction of each cluster; and (iii) Transformation of the data clusters into a multivariate normal distribution. The second part generates the artificial data using the multivariate normal distribution of the first step. The second part generates data for one dimension[1] at a time, starting from the first to the last. It samples the first dimension's data from the normal distribution of the first dimension of the first part's multivariate normal distribution. For the second to last dimensions, it uses a multilayer perceptron (MLP) model that learns the correlations between the current dimension and the first one using the first part's data. It uses the MLP to predict the values for the remaining dimensions of data. It also transforms the artificial data back to the space of the given real data.

We conducted experiments on a popular CSI benchmarks [4] to demonstrate the efficiency of the proposed method. We also compared the performance of the proposed method with two KDE methods. We used two types of tests to measure the performance of the proposed method. The first test uses the Kendall correlation coefficient to find the similarity between the artificial and real samples. The second test is employed to calculate the number of local extrema in

[1]Please note that in the context of this article, the terms variables, dimensions and columns of the data have the same meaning.

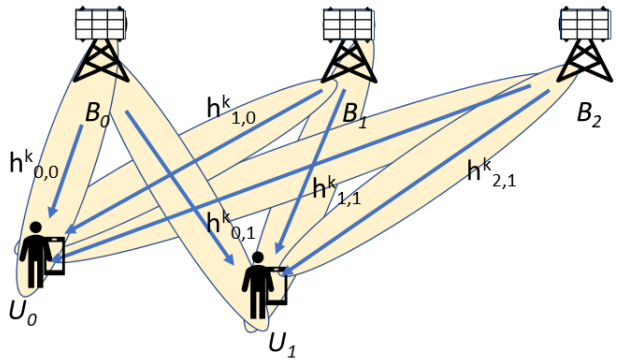


FIGURE 1. A mmWave network of three SBSs and two UEs.

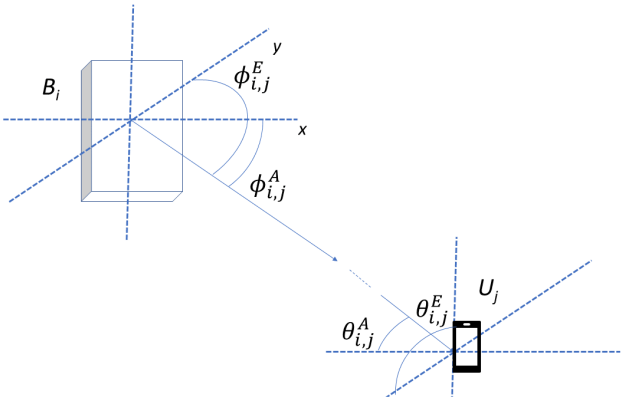


FIGURE 2. Illustration of the AoD and AoA between a pair of SBS and UE.

the samples to detect outliers' presence. The outliers cause non-smoothness in the samples. The results indicate that the proposed method outperforms the KDE-based methods by generating data samples of high similarity with the real data and also is free of outliers that cause non-smoothness.

This article is organized as follows. In Section II we present the relevant background information. In Section III the proposed method is presented. Section IV contains the experimental results in detail. The last section contains the conclusion.

II. BACKGROUND

A. CHANNEL STATE INFORMATION

We consider a MIMO mmWave communications network that employs orthogonal frequency division multiplexing (OFDM). The network consists of N user-equipments (UEs), M SBSs, and K subcarriers. The SBSs use the mmWave links to transmit data to the UE. We denote the set of UE with $\mathcal{U} = \{U_1, U_2, \ldots, U_N\}$, and the set of SBSs with $\mathcal{B} = \{B_1, B_2, \ldots B_M\}$. We assume that each SBS is equipped with a uniform planar array (UPA) based antenna that has $A_1 \times A_2$ antenna elements with uniform spacing of d, and each user has a single antenna element. Fig. 1 shows a mmWave network in which three SBSs transmit data to two UEs by using directed beams. The signals from the SBS reaches the UE following up to L number of multipaths. Multipaths exist because the objects present in the environment cause the signals to go through different degrees of reflection, refraction, and scattering before reaching the UE. The figure also shows the channels between each pair of SBS and UE, and $h^k_{i,j}$ denotes the block-fading channel between the SBS B_i and UE U_j for the subcarrier k.

As already mentioned, for each pair of SBS and UE, the channel consists of up to L paths, and for each path, we have an angle-of-departure (AoD) at the SBS and an angle-of-arrival (AoA) at the UE. Both AoD and AoA have azimuth and elevation components. For B_i and U_j, the azimuth and elevation AoD at the SBS are given by $\phi^A_{i,j}$, $\phi^E_{i,j}$, respectively. The azimuth and elevation AoA at the U_j can be denoted by $\theta^A_{i,j}$, and $\theta^E_{i,j}$, respectively. Fig. 2 illustrates the azimuth and elevation angles. The channel gain $h^k_{i,j}$ is expressed as follows:

$$h^k_{i,j} = \sum_{l=0}^{L-1} \sqrt{\frac{p^{i,j}_l}{K}} e^{\mu^{i,j}_l + \frac{2\pi k}{K}\tau^{i,j}_l B} a_i(\phi^A_{i,j}, \phi^E_{i,j}) \tag{1}$$

In the above equation, $p^{i,j}_l$, $\mu^{i,j}_l$, and $\tau^{i,j}_l$ denote the power, phase, and propagation delay of the signal that follows path l. The symbol B denotes the bandwidth of the signal. The function $a(\phi^A_{i,j}, \phi^E_{i,j})$ denotes the array response vector of the SBS B_i and is given by:

$$a(\phi^A_{i,j}, \phi^E_{i,j}) = a_z(\phi^E_{i,j}) \otimes a_y(\phi^A_{i,j}, \phi^E_{i,j}) \otimes a_x(\phi^A_{i,j}, \phi^E_{i,j}) \tag{2}$$

where, a_x, a_y, and a_z denote the array response vectors in the x, y, and z directions, and their values are given as follows:

$$a_x(\phi^A_{i,j}, \phi^E_{i,j}) = [1, e^{\hat{j}kdsin(\phi^E_{i,j})cos(\phi^A_{i,j})}, \ldots, e^{\hat{j}kd(A_x-1)sin(\phi^E_{i,j})cos(\phi^A_{i,j})}]^T \tag{3}$$

$$a_y(\phi^A_{i,j}, \phi^E_{i,j}) = [1, e^{\hat{j}kdsin(\phi^E_{i,j})sin(\phi^A_{i,j})}, \ldots, e^{\hat{j}kd(A_y-1)sin(\phi^E_{i,j})sin(\phi^A_{i,j})}]^T \tag{4}$$

$$a_z(\phi^E_{i,j}) = [1, e^{\hat{j}kdcos(\phi^E_{i,j})}, \ldots, e^{\hat{j}kd(A_z-1)cos(\phi^E_{i,j})}]^T \tag{5}$$

In the above equations, $\hat{j}$ denotes the imaginary number, and $[..]^T$ denotes the transpose operation. The parameters in the above equations (e.g., AoD and AoA angles, propagation delays of the signals, etc.,) depend on the multi-dimensional environment geometry, materials, locations of SBSs and UEs, etc [4]. Therefore, we need to rely on accurate ray-tracing simulations to provide the necessary data. It is impossible to generate new CSI data samples without any accurate ray-tracing software. In this work, we used a dataset of real CSI samples, denoted by (R_i), and is given as follows.

$$R_i = \begin{bmatrix} h^1_{i,1} & h^2_{i,1} & h^3_{i,1} & \ldots & h^K_{i,1} \\ h^1_{i,2} & h^2_{i,2} & h^3_{i,2} & \ldots & h^K_{i,2} \\ \ldots & & & & \\ h^1_{i,N} & h^2_{i,N} & h^3_{i,N} & \ldots & h^K_{i,N} \end{bmatrix} \tag{6}$$

The dataset R_i contains the CSI samples belonging to the SBS B_i, and any element $h^k_{i,j}$ indicates the complex CSI value of the k^{th}-subcarrier between the SBS B_i and UE U_j. In the above equation, $h^k_{i,j}$ is a complex number and is given by $\{x_1+\hat{j}y_1, x_2+\hat{j}y_2, \ldots, x_K+\hat{j}y_K\}$, where $x_m, y_n \in \mathbb{R}$, for $m, n \in \{1 \ldots K\}$. In this work, we represent a complex number as a concatenation of the real and imaginary parts as follows: $h^k_{i,j} = \{x_1, x_2, \ldots, x_K, y_1, y_2, \ldots., y_K\}$.

III. PROPOSED METHOD

This section contains the description of the proposed method. Fig. 3 shows an overview of the proposed method. The input is a real CSI dataset, and the output is the artificial CSI dataset. The real dataset is denoted by R_i, and refers to the CSI samples of the SBS B_i. The first step is to cluster the given dataset R_i into B clusters based on similarity between the samples. The steps following the initial clustering analyze and synthesize artificial data in a cluster-by-cluster manner. The second step is to apply the principal component analysis (PCA)-based dimensionality reduction to transform the data from $2K$ to K' dimensions, where $K' < 2K$. The third step is to transform the distribution of the data to multivariate normal distribution through the application of probability integral transform (PIT) transformations. The fourth step applies a procedure to generate the artificial data using an MLP-based regression model. The fifth step transforms the artificial data into the same space as the original data through the application of PIT transformations. We discuss each step in detail in the remaining part of this section.

A. CLUSTERING OF DATA

An essential step of the proposed method is clustering that enhances the PCA [11], [12] to successfully reduce the dimensions of the data with minimum information loss. We employed the PCA method for the dimensionality reduction whose information loss is controlled by the percentage of variance explained by the selected number of principal components (i.e., number of dimensions after transformation) [13]. In large-size datasets, a fewer number of principal components explains a small percentage of the total variance, and hence it is impossible to reduce the dimensions of huge size data-sets to a smaller value with a minimal information loss.

The PCA method can reduce the dimensions to a greater extent where there exists a similarity between the samples. Therefore, we divide the given dataset into clusters of relatively similar CSI signals and reduce the dimensions of each cluster separately using the PCA method. The CSI signals have complex shapes, therefore we propose to first transform the signals into a simplified notation that retains the following three features of the data: (i) Number of local extrema (maxima or minima); (ii) Locations of the extrema; and (iii) Type of extrema as either maxima or minima. We empirically found that clustering using these three features can significantly enhance the performance of the PCA method as compared to using random clustering. The proposed clustering method is described formally in the remaining part of this sub-section.

TABLE 1. Description of important variables and parameters.

Symbol	Description
R_i	The dataset of real CSI samples for the SBS B_i
S_i	A transformation of R_i used in the data clustering step
$R_{i,j}$	The j^{th} cluster of the dataset of real CSI samples for the SBS B_i
$R^*_{i,j}$	Projection of $R_{i,j}$ which has $2K$ dimensions into subspace of K' dimensions (where $K' < 2K$)
$R^N_{i,j}$	Transformation of $R^*_{i,j}$ into multivariate normal distribution
$R_{i,j}[k]$, $R^N_{i,j}[k]$	The k^{th} columns of $R_{i,j}$, and $R^N_{i,j}$, respectively.
$\mu_{i,j}[k]$	Mean of $R_{i,j}[k]$
A_i	The artificial dataset of CSI samples of the SBS B_i
$A_{i,j}$	The j^{th} cluster of the artificial data of CSI samples of the SBS B_i
$b_{i,j}$	Number of samples in the data-clusters $R_{i,j}$ or $A_{i,j}$
$A^N_{i,j}$	$A_{i,j}$ after being transformed into having multivariate normal distribution
$A^N_{i,j}[k]$	The k^{th} column of $A^N_{i,j}$
B	Number of clusters
K	Number of subcarriers. The Dimensions of R_i, $R_{i,j}$, A_i, $A_{i,j}$ is equal to $2K$.
K'	Dimensions of the datasets or clusters: $R^N_{i,j}$, $R^*_{i,j}$, $A^N_{i,j}$, $A^*_{i,j}$
M_R	MLP-based regressor with one hidden layer and activation function equal to ReLU
h_l	Number of units in the hidden layer of M_R
$C_{i,j}$	Covariance matrix of $R_{i,j}$
$\lambda_{i,j}$	The Eigenvalues of $C_{i,j}$ is a descending order
$W_{i,j}$	Eigenvectors of $C_{i,j}$ in an order in which the i^{th} column contains the Eigenvector corresponding to the Eigenvalue at the i^{th} position in $\lambda_{i,j}$
I_m	Maximum iterations of the K-means method used in the data clustering step

The proposed clustering method consists of two steps: (i) Application of the data transformation and (ii) Clustering of the transformed data using the K-Means method. The data transformation step contains a transformation function which is applied individually to each channel gain value, i.e., $h^k_{i,j}$ that indicates the channel gain of the k^{th} subcarrier of the channel between the SBS B_i and user u_j. The transformation function for $h^k_{i,j}$ is shown-below, and also requires the channel gain of the $k-1^{\text{th}}$ and $k+1^{\text{th}}$ subcarriers.

$$\begin{aligned}\Pi^k_{i,j} &= \mathcal{T}_c(h^k_{i,j}|\{h^{k-1}_{i,j}, h^{k+1}_{i,j}\}) \\ &= \begin{cases} +1, & \text{if } h^{k-1}_{i,j} < h^k_{i,j} > h^{k+1}_{i,j} \\ -1, & \text{if } h^{k-1}_{i,j} > h^k_{i,j} < h^{k+1}_{i,j} \\ 0, & \text{else} \end{cases}\end{aligned} \tag{7}$$

We apply the function $\mathcal{T}_c(..)$ to all elements of R_i, and denote the resulting transformed form with S_i, as follows.

$$S_i = \begin{bmatrix} \Pi^1_{i,1} & \Pi^2_{i,1} & \ldots & \Pi^K_{i,1} \\ \Pi^1_{i,2} & \Pi^2_{i,2} & \ldots & \Pi^K_{i,2} \\ \ldots & & & \\ \Pi^1_{i,N} & \Pi^2_{i,N} & \ldots & \Pi^K_{i,N} \end{bmatrix}$$

The next step is to apply the K-Means method to partition the complete dataset (R_i) into clusters of smaller datasets, denoted by $R_{i,j}$, where $j \in \{1, 2, .., B\}$ is the index of the

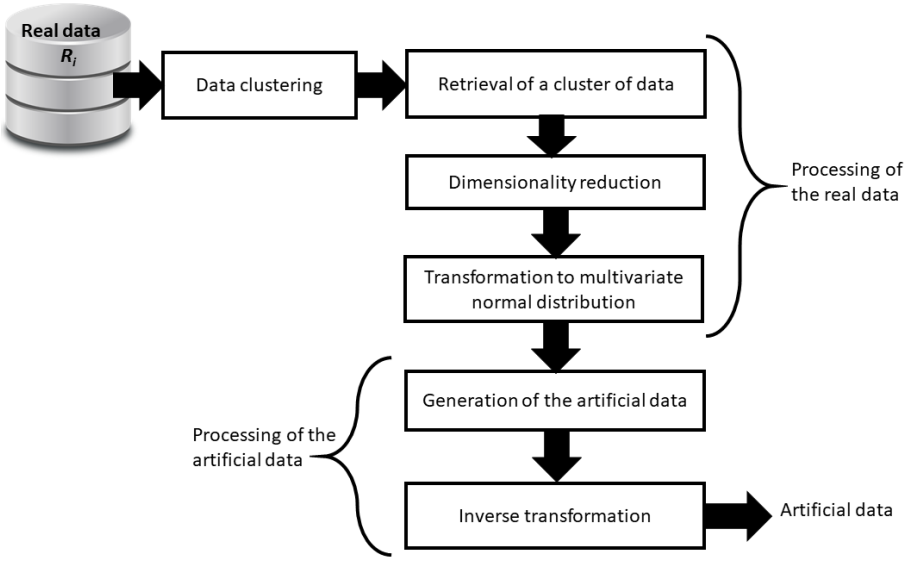


FIGURE 3. Overview of the proposed method.

Algorithm 1: Algorithm to Partition the Real Data Samples Into B Clusters

1 **Input:** R_i real dataset, S_i transformed dataset, $I_m \in \mathbf{Z}^+$;
2 Initialize the cluster centroids $\{\mu_1, \mu_2, \ldots, \mu_B\}$;
3 Initialize $R_{i,j} = \phi$, for $j = 1..B$ and set $i_t = 0$;
4 **while** $i_t < I_m$ **do**
5 **for** *each* $\Pi^k_{i,j} \in S_i$ **do**
6 $c^k_{i,j} = \underset{x \in B}{argmin} ||\Pi^k_{i,j} - \mu_x||^2$;
7 **end**
8 **for** $x \in B$ **do**
9 $\mu_x = \frac{\sum_{i=1}^{M} 1\{c^k_{i,j}=x\}\Pi^k_{i,j}}{\sum_{i=1}^{M} 1\{c^k_{i,j}=x\}}$
10 **end**
11 $i_t = i_t + 1$;
12 **end**
13 **for** $j = 1$ *to* N **do**
14 **for** $k = 1$ *to* K **do**
15 $v = c^k_{i,j}$;
16 $R^k_{i,v} = R^k_{i,v} \cup h^k_{i,j}$;
17 **end**
18 **end**
19 **Output:** $\{R^1_{i,j}, R^2_{i,j}, \ldots, R^K_{i,j}\}$

cluster, and B the number of clusters. Algorithm 1 shows the procedure to cluster the data samples. The algorithm consists of two separate loops. The first 'while' loop performs the clustering using the transformed dataset S_i. The second 'for' loop uses the results of the clustering of the first step to cluster the dataset R_i. The first inner loop (lines 5-7) allocates the sample $\Pi^k_{i,j} \in S_i$ to the cluster whose mean (μ_x) is closest to its value (i.e., having the smallest Euclidean distance), where μ_x denotes the mean of the x^{th} cluster. The second inner loop (lines 8-10) computes the new means for all clusters as equal to the mean of the samples allocated to them. The first part performs clustering using the transformed dataset S_i, however, the clustering results remain valid for the original dataset R_i. The second outer loop (lines 13-18) clusters the samples of dataset R_i using the results of the previous part of the algorithm.

B. DIMENSIONALITY REDUCTION

Dimensionality reduction reduces the number of variables in a cluster of real CSI data (i.e., $R_{i,j}$). This step is important in our work because it enables synthesizing of data with fewer dimensions. The proposed method uses the PCA method, a popular method of dimensionality reduction. Given a data cluster $R_{i,j}$ having B rows and $2K$ columns (i.e., a CSI sample has K real and K imaginary values), the PCA method can reduce the number of columns to K', where $K' < 2K$. In the following, we briefly describe the process of applying the PCA to $R_{i,j}$ as follows:

$$R_{i,j}[k] = R_{i,j}[k] - \mu_{i,j}[k], \quad k \in \{1 \ldots 2K\} \tag{8}$$

$$C_{i,j} = \frac{1}{B-1} R^T_{i,j}.R_{i,j} \tag{9}$$

$$\lambda_{i,j}.W_{i,j} = C_{i,j}.W_{i,j} \tag{10}$$

The above equations use the following notations: $R_{i,j}[k]$ denotes the k^{th} column of $R_{i,j}$; $\mu_{i,j}[k]$ denotes the mean of $R_{i,j}[k]$; $C_{i,j}$ is the covariance of $R_{i,j}$ which can be obtained by taking the dot product of $R_{i,j}$ and its transpose ($R^T_{i,j}$); $\lambda_{i,j}$ and $W_{i,j}$ denote the Eigenvalues and Eigenvectors of $C_{i,j}$, respectively. We call $W_{i,j}$ as principal components of $R_{i,j}$. We assume that $\lambda_{i,j}$ contains Eigenvalues in a descending order and $W_{i,j}$ also contain Eigenvectors following the order of Eigenvalues in $\lambda_{i,j}$, i.e., the first column of $W_{i,j}$ contains the

Eigenvector corresponding to the largest Eigenvalue in $\lambda_{i,j}$. To reduce the dimensions of data, we need to keep the first K' Eigenvectors in $W_{i,j}$. Mathematically, the dimensionality reduction can be expressed as follows:

$$W^*_{i,j} = W_{i,j}[, 1 : K'] \tag{11}$$
$$R^*_{i,j} = R_{i,j} W^*_{i,j} \tag{12}$$

In the above expression, $R^*_{i,j}$ denotes the data ($R_{i,j}$) projected to the subspace of K' dimensions (where $K' < 2K$). The dimensions of $R^*_{i,j}$ is equal to ($b_{i,j} \times K'$), where $b_{i,j}$ denotes the number of samples in the cluster $R_{i,j}$. An important design decision is to choose a suitable value of K' for a given batch size B. We choose the value of K' by considering the explained variance of each principal component. The explained variance is often used in statistics to express the model's accuracy, and its value can lie between 0 and 1, and higher values are considered better. We use the symbol ν_i to denote the explained variance of the i^{th} principal component (where the i^{th} principal component is equal to the Eigenvector at the i^{th} column in $W_{i,j}$) is given by:

$$\nu_i = \frac{\sqrt{\lambda_i}}{\sum_{i=1}^{2K} \sqrt{\lambda_i}} \tag{13}$$

We can minimize the information loss in reducing the dimensions of the data by choosing a value of K' which is a compromise between being a small and a high value of percentage of explained variance, i.e., $\sum_{i=1}^{K'} \nu_i$. K' is considered a user-defined parameter in this work, and the experimental results section discusses the observations relevant to choosing appropriate values of the parameters including B and K'.

C. TRANSFORMATION TO NORMAL DISTRIBUTION

We apply the PIT method [14] to transform the distribution of each column of data to normal. We should independently apply the PIT method to each dimension of data. The transformation procedure consists of the following steps. Let us consider the k^{th} column of the data-cluster, denoted by $R^*_{i,j}[k]$ whose PDF is f_R, and cumulative distribution function (CDF) is $\mathbb{F}_R$. The transformation of $R^*_{i,j}[k]$ comprises the following steps.

$$U_{i,j}[k] = \mathbb{F}_R(R^*_{i,j}[k]) \tag{14}$$
$$R^N_{i,j}[k] = \mathbb{F}^{-1}_N(U_{i,j}[k]) \tag{15}$$

The above equations show the two steps. The first step is to transform $R^*_{i,j}$ using its own CDF that returns a uniform distribution $U_{i,j}[k] \sim U(0, 1)$. The second step is to transform the data obtained in the first step by the inverse CDF of the standard normal distribution ($N(0, 1)$), which is also known as the Quantile function. The data obtained after applying inverse CDF transformation to $U_{i,j}[k]$ has the normal distribution, denoted by $R^N_{i,j}[k]$. The quantile function, denoted $\mathbb{F}^{-1}_N$, does not have a closed-form algebraic equation, and it should be determined by approximation. Currently, almost all statistical softwares contain the implementation of the quantile function of the normal distribution. Readers can refer to the documentation of individual tools for further details, e.g. 'norm.ppf' function in Python's SciPy package [15].

D. GENERATION OF ARTIFICIAL DATA

The task of synthesizing the artificial data is to generate a multivariate normal distribution of K' variables with correlations between different variables (or columns). The proposed work sequentially generates the data starting from the first to the last column. We sample data from a normal distribution to generate the first column, and use an MLP regression model to generate data for the remaining columns. The remaining part of this section describes the algorithms for the generation of artificial data.

Algorithm 2: Algorithm to Generate the First Column of the Artificial Data

1 **Input:** $R^N_{i,j}[1]$: First column of the data-cluster $R^N_{i,j}$
$\zeta_m = \mathbb{E}(R^N_{i,j}[1])$;
2 $\zeta_s = \sqrt{\mathbb{E}(R^N_{i,j}[1] - \zeta_m)^2}$;
3 Generate an array of data, $X \in U(0, 1)$;
4 Apply transformation $A^N_{i,j}[1] = \mathbb{F}^{-1}_N(X, \zeta_m, \zeta_s)$;
5 **Output:** $A^N_{i,j}[1]$

Fig. 2 shows the method to generate the first column of the batch of the artificial data, denoted by $Ai, j^N[1]$. The input is $R^N_{i,j}[1]$ (i.e., the first column of $R^N_{i,j}$), and the output is the first column of the artificial data, $A^N_{i,j}[1]$. The method first computes the mean (ζ_m) and standard deviation (ζ_s) of $R^N_{i,j}[1]$. Assuming that the artificial data's batch size is equal to the real dataset, the remaining part of the algorithm generates random real numbers following the normal distribution using the inverse CDF method. In the inverse CDF method, we first sample random numbers X from a uniform distribution $U(0, 1)$, and then transform the generated data using the inverse-CDF of the normal distribution, $\mathbb{F}^{-1}_N(\zeta_m, \zeta_s)$ [15]. The length of X can be same as the length of $R^N_{i,j}[1]$ which is denoted by $b_{i,j}$.

Algorithm 3: Algorithm to Generate the Second to K' Columns of the Artificial Data

1 **Input:** $A^N_{i,j}[1]$, $R^N_{i,j}$;
2 **for** $i = 2$ *to* K' **do**
3 Train the MLP regressor (M_R) with data $\{R^N_{i,j}[1], R^N_{i,j}[i])\}$;
4 Using M_r, predict the value of $A^N_{i,j}[i]$ by giving input $A^N_{i,j}[1]$ to M_R
5 **end**
6 **Output:** $A^N_{i,j}[2] \ldots A^N_{i,j}[K']$

Algorithm 3 shows the method to generate the second to K'^{th} columns of the artificial data ($A^N_{i,j}[2] \ldots [K']$). The input

are the batch of real dataset ($R^N_{i,j}$), and the first column of the batch of artificial data ($A^N_{i,j}[1]$). The method uses an MLP-model to predict the values of the new columns using the values of the first column, i.e., we use the MLP to predict the columns 2 to K' using the data of the first column. For each column (where, k indicates the index of the column), the prediction consists of two steps: (i) Training of the MLP model (M_R) using the data of the two columns $R^N_{i,j}[1]$, and $R^N_{i,j}[k]$; and (ii) Using M_R to predict the values when the input to it is $A^N_{i,j}[1]$. The prediction returns the values of $A^N_{i,j}[k]$, i.e., the k^{th} column of the data-cluster of the artificial data. The MLP used in the regression consists of single input and output with a hidden layer of l_m number of neurons. The activation function is rectified linear unit (ReLU). The training of the MLP uses two columns of the real dataset $R^N_{i,j}[1]$ and $R^N_{i,j}[k]$, where k is the index of the column whose data we need to generate. Up-to v_r% of the data is used as a validation set and the remaining as the training set in each batch. The training also uses regularization, and the weight-decay parameter of L2-regularization is denoted by α_r. The learning-rate that controls the amount of change in the value of parameters is denoted by α_l. The training of weights is accomplished using Adam [16], which is a popular stochastic gradient-based algorithm. The convergence criterion of the training is considered equal to the occurrence of up-to n_r number of iterations with an improvement less than t_r in the value of the loss function, and the training always terminates after T_r number of iterations.

E. INVERSE TRANSFORMATIONS

The generated artificial data have a normal distribution and are in the K'-dimensional space. The PIT method describes the method to transform the data in any distribution to the standard uniform distribution ($U(1, 0)$) by transforming it from its CDF. Similarly, the data with a standard uniform distribution can be transformed to any other distribution by employing the inverse-CDF of the target distribution. In this work, we apply three transformations to the artificial data to project it to the same space as the input CSI data, and transformations are applied independently to each column. Below we discuss the procedure to obtain the k^{th} column of the artificial data, denoted by $A^N_{i,j}[k]$. The first transformation in $\mathbb{F}_N$, i.e., the CDF of the normal distribution to transform the data to the uniform distribution. The second transformation is to apply the inverse CDF of the $R^*_{i,j}[k]$, denoted by $\mathbb{F}^{-1}_R$. The third transformation is to multiply by $W^*_{i,j}$ to project the data back to the space of K-dimensions. The mathematical expressions denoting the application of three transformations on $A^N_{i,j}[k]$ are given below.

$$A^*_{i,j} = \mathbb{F}_N(A^N_{i,j}) \tag{16}$$

$$A^*_{i,j} = \mathbb{F}^{-1}_R(A^*_{i,j}) \tag{17}$$

$$A_{i,j} = A^*_{i,j}.(W^*_{i,j})^T \tag{18}$$

TABLE 2. Parameters used in dataset generation.

parameter	value
Number of SBSs	2 $\{B_3, B_4\}$
Rows containing users	1000 to 1020
Number of antennas in the UPA of the SBSs	1 x 8 x 8
Total number of sub-carriers	1024
Number of subcarriers included in the CSI data collection	64
Number of paths between the antenna and UE	5

TABLE 3. Parameters of the proposed method.

parameter	value
K'	3
B	500
l_m	200
v_r	0.10
α_r	0.0001
α_l	0.001
n_r	10
t_r	0.0001
I_m	300

In the above equations, $\mathbb{F}_\mathbb{N}$ refers to the CDF function of the standard normal distribution, and $\mathbb{F}_\mathbb{R}{}^{-1}$ refers to the inverse CDF of $R_{i,j}$. The last equation indicates the transformation from the K' to K dimensional space. Please note that the dimensions of $A^*_{i,j}$, and $W^*_{i,j}$ are equal to $(b_{i,j} \times K')$, and $(2K \times K')$, and the dot-product of $A^*_{i,j}.(W^*_{i,j})^T$ project the artificial data to the $2K$-dimensional space.

IV. EXPERIMENTAL RESULTS

This section describes the experimental results conducted to demonstrate the performance of the proposed architecture.

In this work, we use the mmWave dataset, 'DeepMIMO' [4]. We configure the dataset to use raytracing data of the scenario "O1" with a 60 GHz operating frequency. The scenario "O1" is an outdoor model and consists of two streets and one intersection. The SBS lie along the roads' sides, and the UEs lie uniformly distributed on the streets. The model also contains buildings of varying heights that affect the path of the signals. Table 2 shows the critical parameters used in the dataset generation. Please note that the parameters not specified in Table 2 maintain their default values.

In the selection of the parameter values of the proposed method, a critical decision is to choose the B and K' values. These two values control the amount of information loss in reducing the dimensions of the given real dataset. Increasing the number of clusters, i.e., the value of B enables us to choose a smaller K' while ensuring minimal information loss. In this work, we want to keep K' to a smaller value to prevent the generation of high-dimensional artificial data. We measure the information loss of a data sample as the Euclidean distance between a recovered data sample and its original data sample, where recovered data is the one which is inverse transformed to the original $2K$ dimensions after

being reduced to K' dimensions. Fig. 4 shows the relationship between the values K', B, and information loss using the dataset of B_3. The results show that for $K' = 3$ and $B = 500$, the upper, middle, and upper quartiles of the loss values are: 0.0566, 0.124, and 0.1989, respectively. For $K' = 4$, and $B' = 500$, the lower, middle and upper quartiles of the loss values are: 0.033, 0.077, and 0.131, respectively. The loss values for the remaining pairs of K' and B are higher than these two cases. Therefore, we used $K' = 3$ and $B = 500$ in this work.

The mmWave networks contain a large number of users visible to each SBS, a large number of subcarriers, and multi-paths between the SBS and users [4]. Therefore, the CSI dataset of even one SBS has a high number of rows and columns, i.e., it is both high dimensional and has many samples of different shapes. We need to reduce the dimensions of the data to simplify the problem of the generation of synthetic data. A challenge in dimensionality reduction is to keep the information loss to a small value. We propose a data clustering method that uses an innovative data transformation and enable us to cluster the CSI dataset for dimensionality reduction with very small information loss. Here, we demonstrate the benefit of applying the proposed clustering method by comparing it against two alternative approaches of doing clustering: random clustering, and no-clustering. In random clustering, we randomly divide the samples into B number of clusters, and in the no-clustering approach we do not divide the data. Fig. 5 shows that the mean information loss in reducing the dimensions while employing: (i) the proposed data method, (ii) random-clustering, and (iii) no-clustering. The plot shows the results of information loss in applying dimensionality reduction with the number of reduced dimensions equal to three ($K' = 3$) to the dataset of B_3 that has 50K samples, and 128 columns ($K = 128$). The results show that the proposed clustering method has an information loss of only 0.14, whereas the alternative approaches suffer a very high information loss that lies between 0.5–1.29. The significant loss of information results in the generation of distorted signals. Fig. 6 shows the CSI samples generated using the three approaches of doing clustering. The figure shows that the samples generated using random and no-clustering are distorted, and the sample generated using the proposed method is not distorted. These simulations suggest that the proposed clustering method can greatly improve the dimensionality reduction process and hence improve the quality of the artificial data.

To compare the performance of the proposed method with existing ones, we also implemented the KDE method which is one of the popular methods for the generation of artificial data. A crucial choice in the implementation of the KDE method is the value of the bandwidth parameter. We have two popular rules of thumbs for choosing the bandwidth value, given as follows:

$$h_1 = 1.06\tau_j N^{\frac{1}{5}} \tag{19}$$

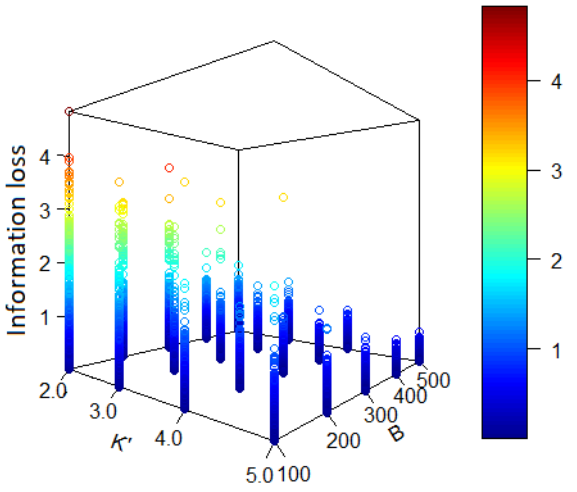


FIGURE 4. Relationship among B, K' and information loss.

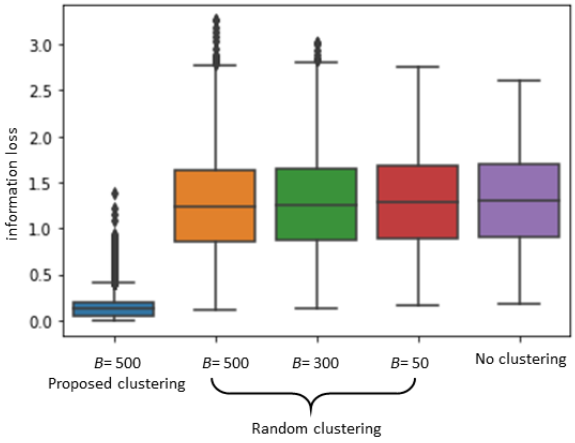


FIGURE 5. The information loss in applying the PCA based dimensionality reduction ($K' = 3$) using the proposed clustering technique, random clustering with different number of clusters and without any clustering.

$$h_2 = 0.9 \min(\tau_j \frac{I_j}{1.34}) n^{-\frac{1}{5}} \tag{20}$$

In the above expression, τ_j, and I_j denote the standard deviation (SD) and interquartile range (IQR) of the j^{th} column of the data-set. The expression in Equation 19 of h_1 is called as Scott's rule, and the formula for h_2 is known as Silverman's rule. The Scott's rule of thumb has a limitation that the input data should have a normal distribution. We denote the implementation of the KDE methods with the Scott's rule and Silverman's rule as KDE1, and KDE2, respectively. In both KDE1, KDE2, we used the quantile transformation (QT) to transform the input data to normal distribution.

Our aim in this work is to generate synthetic data which is similar to real data. To accomplish this, we used Kendall correlation coefficient [17], [18], denoted by τ_D, and detection of outliers that cause non-smoothness in the samples. The Kendall coefficient is used to determine the similarity

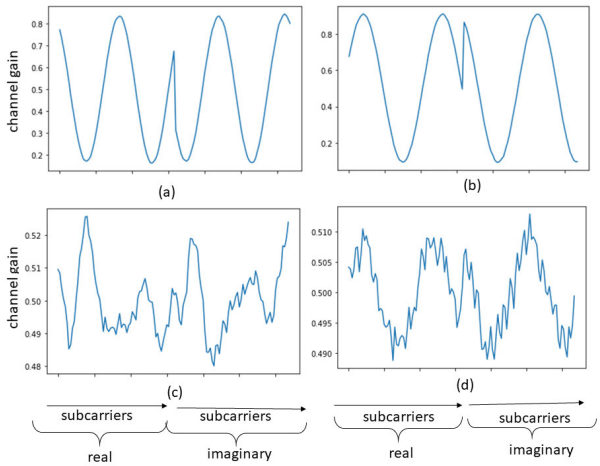


FIGURE 6. Illustration of original and artificial CSI samples,(a) Original; (b) Generated with the proposed clustering technique ($B = 500$); (c) Generated using Random clustering ($B = 500$); and (b) Generated without any clustering.

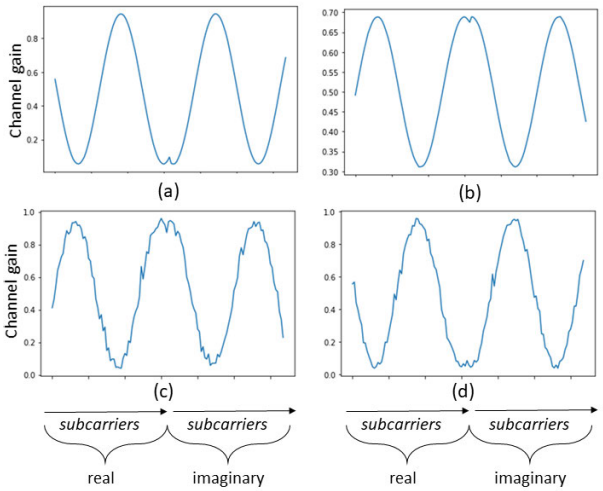


FIGURE 7. Illustration of a real CSI data sample of SBS B_3 and its closest-shape artificial signal generated by the proposed and existing methods: (a) Real data; (b) Proposed method ($\tau_K = 0.98$); (c) KDE_1, ($\tau_K = 0.94$); and (d) KDE_2, ($\tau_K = 0.95$).

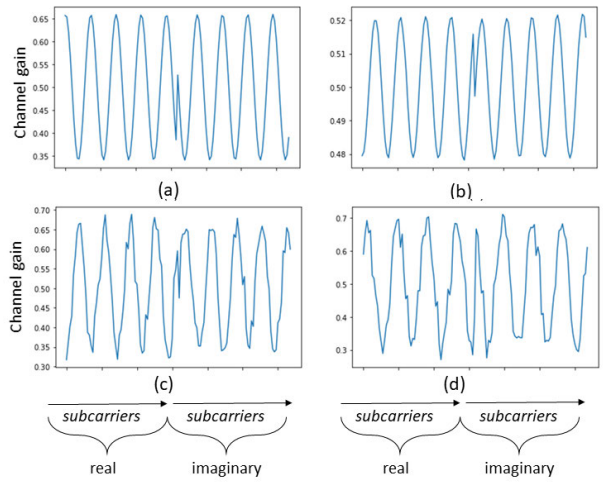


FIGURE 8. Illustration of a real CSI data sample of SBS B_4 and its closest-shape artificial signal generated by the proposed and existing methods: (a) Real data; (b) Proposed method ($\tau_K = 0.97$); (c) KDE_1, ($\tau_K = 0.84$); and (d) KDE_2, ($\tau_K = 0.87$).

between data samples, and the number of outliers given the non-smoothness in the samples. We detect outliers by finding extrema points [15] in the samples. Further details are discussed in subsections below.

A. USING KENDALL CORRELATION COEFFICIENT

The Kendall coefficient (τ_D) is often used in signal processing to determine the similarity between two signals. The values of τ_D lies between 0-1 and the two signals are closer when they have a value closer to one. This section presents the results of applying τ_D between the real dataset and the artificial data generated by the proposed method, KDE1, and KDE2 methods. The τ_D of each signal is computed by finding its minimum values with all signals present in the real dataset. The x-axis contains the index and the y-axis contains the values of the attributes CSI of CSI samples. We want to recall that a CSI sample contains K complex numbers, and our representation of the CSI samples consists of $2K$ attributes, where the first half indices contain the real components and the last half contains the imaginary components. As examples, Figs. 7-8 show the real and artificial samples, and the captions indicate the τ_D values of the samples. The examples show that the proposed method's artificial samples look similar to the real ones and have high τ_D values. The remaining part of this section presents the results of the τ_D computation for the proposed method, and for KDE1 and KDE2 methods.

The histograms in Figs. 9-10 show that in general the artificial CSI samples of the proposed method have higher τ_K values as compared to KDE1 and KDE2 for both SBSs (B_3 and B_4). Fig. 9 shows that the artificial generated by the proposed method for the SBS B_3 have the lower quartile (Q1), upper (Q3) quartile and, mean (Q2) equal to 0.975, 0.986, and 0.990, respectively. For the KDE1 and KDE1 methods, the values of the three quartiles (Q1, Q2, Q3) are as follows: 0.874, 0.905, and 0.928, respectively, for the KDE1 method, and 0.892, 0.892, and 0.916, respectively, for KDE2 method. For the SBS B_4, the values of the Q1, Q2, and Q3 quartiles of the proposed method are as follows: 0.942, 0.964, and 0.984, respectively. The values of the Q1, Q2, and Q3 quartiles of the KDE1 method are as follows: 0.878, 0.909, and 0.931, respectively. The values of the Q1, Q2, and Q3 quartiles for the KDE2 method are as follows: 0.903, 0.921, and 0.941, respectively.

This subsection shows that the proposed method's signals have a much closer resemblance to the real signals than the KDE1 and KDE2 methods' signals.

B. DETECTING OUTLIERS IN THE SAMPLES

An artificial sample could contain some points known as outliers that deviate from the assumed values and make the

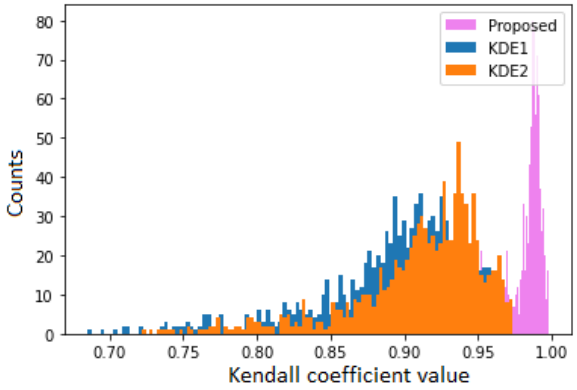


FIGURE 9. Histograms showing the Kendall coefficient (τ_K) values of the proposed and existing methods for the SBS B_3.

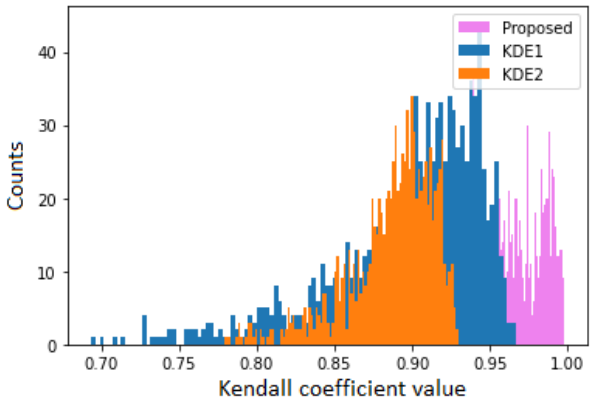


FIGURE 10. Histograms showing the Kendall coefficient (τ_K) values of the proposed and existing methods for the SBS B_4.

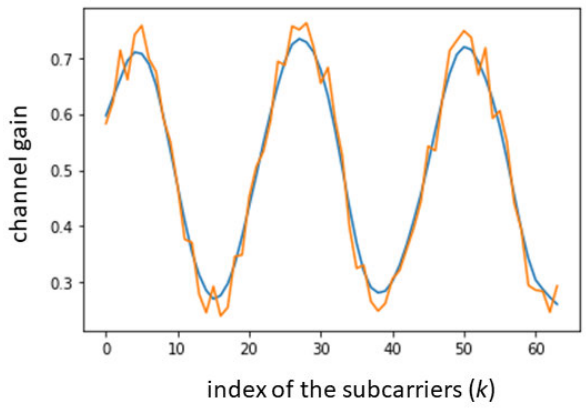


FIGURE 11. An artificial sample generated by the KDE1 method along with its version after smoothness.

signal appear non-smooth or rough. The outliers are a result of the inaccurate generation of values. As an example, Fig. 11 shows a sample of the KDE1 method (in orange) with its smoothed version (in blue). The sample has outliers that cause it looks non-smooth. The methods to detect outliers vary

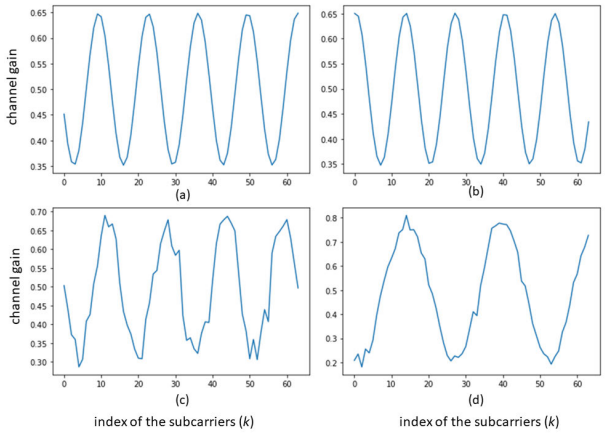


FIGURE 12. CSI samples and their number of local extreme points, (a) real data sample (# of extrema = 9); (b) Artificial data sample of the proposed method (# of extrema = 9); (c) Artificial data sample of the KDE1 method (# of extrema = 20); (d) Artificial data sample of the KDE2 method (# of extrema = 14).

with data types. In this work, any deviation of the values from the smooth curve illustrates local extrema. The local extrema include all those points in a CSI sample for which the value is either minimum or maximum than its neighboring points [15]. We can say that the CSI samples $R_{i,j}$ or $A_{i,j}$ of length equal to K have a local maxima at k under the following conditions:

$$R_{i,j}[k-1] < R_{i,j}[k] > R_{i,j}[k+1] \tag{21}$$
$$A_{i,j}[k-1] < A_{i,j}[k] > A_{i,j}[k+1] \tag{22}$$

Similarly, the CSI samples $R_{i,j}$ or $A_{i,j}$ have a local minima at k, if the following conditions hold.

$$R_{i,j}[k-1] > R_{i,j}[k] < R_{i,j}[k+1] \tag{23}$$
$$A_{i,j}[k-1] > A_{i,j}[k] < A_{i,j}[k+1] \tag{24}$$

Fig. 12 shows the CSI samples belonging to the real dataset and the artificial datasets generated by the proposed methods, namely KDE1 and KDE2. The figure caption also mentions the number of extreme points in the samples. The number of extreme points in the artificial samples of KDE1 and KDE2 methods are much more than the real-dataset because of the unwanted small peaks present in the data. The number of extreme points in the artificial samples of the proposed method is equal to that of the real-data because the sample is smooth. Histograms in Fig. 13 summarize the number of extreme points resulting from all real and artificial samples. The histograms indicate that the real-dataset and the proposed method have almost equal number of extreme points, whereas KDE1 and KDE2 have a high number of extreme points. Table 4 numerically summarizes the number of extremes points results. The table also indicates that the number of extreme points in the artificial samples of the proposed method is quite close to that in real-data, whereas the artificial samples of the KDE1 and KDE2 have a high number of extreme points.

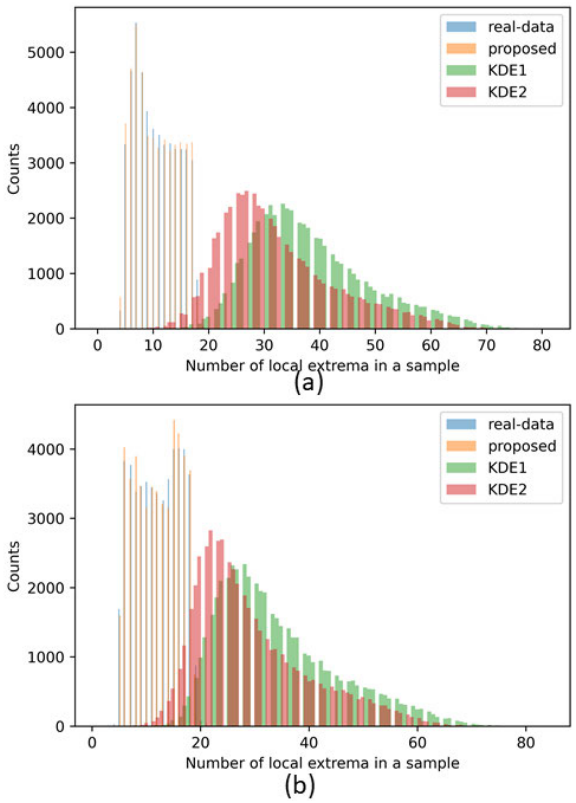


FIGURE 13. Histograms of the number of extrema in the CSI samples of (a) B_3, (b) B_4.

TABLE 4. Number of extreme points in the CSI samples.

SBS	Method	Mean	standard deviation
B_3	real-data	10.62	3.83
	proposed	10.57	3.88
	KDE1	38.16	10.68
	KDE2	32.34	10.55
B_4	real-data	11.98	4.06
	proposed	11.97	4.05
	KDE1	34.20	11.16
	KDE2	29.33	10.58

The results of analyzing the samples using the Kendall coefficient and the number of extrema indicate that the artificial samples generated by the proposed method are similar in shape to the real samples, and also free from outliers. The artificial samples of the KDE1 and KDE2 methods are not similar to the real CSI samples, and have outliers.

V. CONCLUSION AND FUTURE WORK

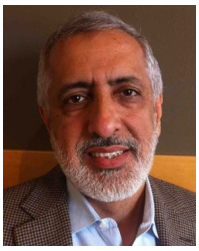
This article presented a fast and simple method to synthesize artificial CSI data of mmWave networks. The CSI data in mmWave networks can only be obtained through physical experimentation and expensive simulations. The proposed work enables the users to extend the given datasets with real-like artificial samples. The proposed method consists of data clustering, several data transformations, data sampling, and prediction using an MLP-based regression method. We compared the proposed method with two variants of the KDE-method. The results indicate that the artificial CSI samples generated by the proposed method have much higher similarity with the real CSI samples compared to the KDE-based methods. Recently, dense MMW networks also started to gain attention. They have a dense deployment of SBSs to minimize the effects of path loss, shadowing, and blockages [19]. Although the abundance of SBSs enables more users to have strong and/or line-of-sight (LOS) channels with the SBSs, it introduces severe interference. The interference in dense mmWave networks can be mitigated by implementing cooperation among the SBS [19]–[21]. In case of cooperating SBSs, the CSI of the SBSs no longer remain independent from each other. The correlation amongst the CSI of different SBSs must be captured and considered in the synthesis of artificial data. The representation of the correlated CSI data of multiple SBSs results in very high dimensional data. The proposed method incorporates data clustering and dimensionality reduction that can be extended further to synthesize CSI data in the presence of cooperation among the SBSs to mitigate the interference problem in dense mmWave networks.

REFERENCES

[1] A. Alizadeh and M. Vu, "Load balancing user association in millimeter wave MIMO networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 6, pp. 2932–2945, Jun. 2019.
[2] P. Paul, H. Wu, and C. Xin, "BOOST: A user association and scheduling framework for beamforming mmWave networks," *IEEE Trans. Mobile Comput.*, early access, May 6, 2020, doi: 10.1109/TMC.2020.2992623.
[3] L. Liu, C. Oestges, J. Poutanen, K. Haneda, P. Vainikainen, F. Quitin, F. Tufvesson, and P. Doncker, "The COST 2100 MIMO channel model," *IEEE Wireless Commun.*, vol. 19, no. 6, pp. 92–99, Dec. 2012.
[4] A. Alkhateeb, "DeepMIMO: A generic deep learning dataset for millimeter wave and massive MIMO applications," in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, USA, Feb. 2019, pp. 1–8.
[5] G. Papamakarios, "Neural density estimation and likelihood-free inference," 2019, *arXiv:1910.13233*. [Online]. Available: https://arxiv.org/abs/arXiv:1910.13233
[6] A. Goncalves, P. Ray, B. Soper, J. Stevens, L. Coyle, and A. P. Sales, "Generation and evaluation of synthetic patient data," *BMC Med. Res. Methodol.*, vol. 20, no. 1, p. 108, May 2020.
[7] J. Unpingco, *Python for Probability, Statistics, and Machine Learning*. Cham, Switzerland: Springer, 2019.
[8] L. M. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for GANs do actually converge?" in *Proc. 35th Int. Conf. Mach. Learn.* Stockholm, Sweden: Stockholmsmässan, 2018, pp. 3481–3490.
[9] X. Ma, R. Jin, K.-A. Sohn, J.-Y. Paik, and T.-S. Chung, "An adaptive control algorithm for stable training of generative adversarial networks," *IEEE Access*, vol. 7, pp. 184103–184114, 2019.
[10] X. Chen, C. Xu, X. Yang, L. Song, and D. Tao, "Gated-GAN: Adversarial gated networks for multi-collection style transfer," *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 546–560, Feb. 2019.
[11] D. C. Lay, *Linear Algebra and Its Applications*, 4th ed. New York, NY, USA: Addison-Wesley, 2012.
[12] U. Chiavetta, B. Lasserre, P. Di Martino, and M. Marchetti, "A simple multivariate analysis to assess diversity in a complex long-term managed forest area in central Italy," *Plant Biosystems-Int. J. Dealing With All Aspects Plant Biol.*, vol. 149, no. 6, pp. 1015–1024, Nov. 2015, doi: 10.1080/11263504.2014.980358.
[13] I. T. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Phil. Trans. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 374, no. 2065, Apr. 2016, Art. no. 20150202, doi: 10.1098/rsta.2015.0202.
[14] J. E. Angus, "The probability integral transform and related results," *SIAM Rev.*, vol. 36, no. 4, pp. 652–654, Dec. 1994.

[15] P. Virtanen *et al.*, "SciPy 1.0: Fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, pp. 261–272, Feb. 2020.
[16] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, Y. Bengio and Y. LeCun, Eds., San Diego, CA, USA, May 2015, pp. 1–15.
[17] H. Chadwick and L. Kurz, "Rank permutation group codes based on Kendall's correlation statistic," *IEEE Trans. Inf. Theory*, vol. IT-15, no. 2, pp. 306–315, Mar. 1969.
[18] G. Zhao, Y. Zhang, G. Zhang, D. Zhang, and Y.-J. Liu, "Multi-target positive emotion recognition from EEG signals," *IEEE Trans. Affect. Comput.*, early access, Dec. 8, 2020, doi: 10.1109/TAFFC.2020.3043135.
[19] W. Feng, Y. Wang, D. Lin, N. Ge, J. Lu, and S. Li, "When mmWave communications meet network densification: A scalable interference coordination perspective," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 7, pp. 1459–1471, Jul. 2017.
[20] T. Bai, A. Alkhateeb, and R. Heath, "Coverage and capacity of millimeter-wave cellular networks," *IEEE Commun. Mag.*, vol. 52, no. 9, pp. 70–77, Sep. 2014.
[21] D. Maamari, N. Devroye, and D. Tuninetti, "Coverage in mmWave cellular networks with base station co-operation," *IEEE Trans. Wireless Commun.*, vol. 15, no. 4, pp. 2981–2994, Apr. 2016.

UMAIR F. SIDDIQI (Member, IEEE) was born in Karachi, Pakistan, in 1979. He received the B.E. degree in electrical engineering from the NED University of Engineering and Technology, Karachi, in 2002, the M.Sc. degree in computer engineering from the King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, Saudi Arabia, in 2007, and the D.Eng. degree from Gunma University, Japan, in 2013. He is currently pursuing a certification program in machine learning methods with the University of California San Diego Extension, USA. He is also a Research Engineer with the Center of Communications and Information Technology Research, Research Institute, KFUPM. He has authored over 35 research articles in international journals and over 35 research papers in international conferences. He also has several U.S. patents. His research interests include machine/deep learning, deep reinforcement learning, metaheuristic algorithms, and optimization.



SADIQ M. SAIT (Senior Member, IEEE) was born in Bengaluru. He received the bachelor's degree in electronics engineering from Bangalore University, in 1981, and the master's and Ph.D. degrees in electrical engineering from the King Fahd University of Petroleum & Minerals (KFUPM), in 1983 and 1987, respectively. He is currently a Professor of computer engineering and the Director of the Center for Communications and IT Research, Research Institute, KFUPM. He has authored over 200 research articles, contributed chapters to technical books, granted several U.S. patents, and lectured in over 25 countries. He is also the principle author of two books. He received the Best Electronic Engineer Award from the Indian Institute of Electrical Engineers, Bengaluru, in 1981.

KHALED ABDUL-AZIZ AL-UTAIBI (Member, IEEE) was born in Riyadh, Saudi Arabia, in 1973. He received the B.S., M.S., and Ph.D. degrees in computer engineering from the King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, in 1997, 2002, and 2019, respectively. From 2002 to 2004, he worked as a Lecturer with the Department of Computer Engineering, King Fahd University of Petroleum and Minerals. In 2004, he joined the Department of Electrical Engineering, University of Hail, Hail, Saudi Arabia, as a Lecturer, and became an Assistant Professor, in 2020. His current research interests include image cryptography, optimization algorithms, machine learning, and artificial intelligence.