REM-U-net: Deep Learning Based Agile REM Prediction with Energy-Efficient Cell-Free Use Case

Hazem Sallouha[1], Shamik Sarkar[2], Enes Krijestorac[2], and Danijela Cabric[2]

[1]Electrical Engineering Department, KU Leuven, Leuven, Belgium
[2]Electrical and Computer Engineering Department, University of California, Los Angeles, USA

Abstract—Radio environment maps (REMs) hold a central role in optimizing wireless network deployment, enhancing network performance, and ensuring effective spectrum management. Conventional REM prediction methods are either excessively time-consuming, e.g., ray tracing, or inaccurate, e.g., statistical models, limiting their adoption in modern inherently dynamic wireless networks. Deep-learning-based REM prediction has recently attracted considerable attention as an appealing, accurate, and time-efficient alternative. However, existing works on REM prediction using deep learning are either confined to 2D maps or use a limited dataset. In this paper, we introduce a runtime-efficient REM prediction framework based on u-nets, trained on a large-scale 3D maps dataset. In addition, data preprocessing steps are investigated to further refine the REM prediction accuracy. The proposed u-net framework, along with preprocessing steps, are evaluated in the context of *the 2023 IEEE ICASSP Signal Processing Grand Challenge, namely, the First Pathloss Radio Map Prediction Challenge*. The evaluation results demonstrate that the proposed method achieves an average normalized root-mean-square error (RMSE) of 0.045 with an average of 14 milliseconds (ms) runtime. Finally, we position our achieved REM prediction accuracy in the context of a relevant cell-free massive multiple-input multiple-output (CF-mMIMO) use case. We demonstrate that one can obviate consuming energy on large-scale fading measurements and rely on predicted REM instead to decide on which sleep access points (APs) to switch on in a CF-mMIMO network that adopts a minimum propagation loss AP switch ON/OFF strategy.

Index Terms—AP switch ON/OFF, cell-free, deep learning, large-scale fading, pathloss, radio environment map, received signal strength, spatial prediction, u-net

I. Introduction

Predicting radio environment maps (REMs) that captures large-scale fading (LSF) could facilitate deployment planning and operation optimization of wireless networks. For instance, REM prediction can play an essential role in spectrum sharing [1], localization [2], path planning for UAVs [3], finding coverage holes [4], optimizing resource allocation in dense networks [5], etc.

Why not ray tracing? REM prediction using ray-tracing [6] is well-known for its high accuracy, outperforming other conventional statistical model-based methods, such as the 3GPP spatial channel model [7], COST 231 model [8], and WINNER II model [9]. However, the challenge with ray tracing is its prohibitively long computation time, limiting its adoption in modern networks, which are designed to be dynamic in terms of frequent resource allocation, reconfigurability, and mobility. This issue of computation time is especially critical for resource-constrained devices/nodes in distributed wireless networks.

Deep learning vs. ray tracing: Recent research works have explored the use of supervised deep learning, specifically convolutional neural networks (CNNs) as a swift alternative for ray-tracing-based REM prediction [3,10,11]. The basic idea is to use a deep neural network (DNN) as a function that approximates the input-output mapping of ray tracing methods in a faster manner. However, this faster approximation comes at the cost of reduced prediction accuracy. Accordingly, a significant focus of ongoing research in deep learning-based REM prediction is to improve the prediction accuracy while at the same time ensuring a bounded prediction runtime, e.g., in the scale of a few milliseconds. This research goal was also the foundation of the *2023 IEEE ICASSP First Pathloss Radio Map Prediction Challenge*. At the same time, deep learning-based REM prediction has also been shown to be superior to traditional signal processing-based REM prediction [10].

Reactive vs. proactive deep learning REM prediction: Deep learning-based REM prediction, as well as REM prediction in general, can be broadly categorized as reactive and proactive. Reactive REM prediction relies on a small set of RSS measurements from an active transmitter whose radio environment is to be predicted [3]. In contrast, proactive REM prediction is capable of making predictions for transmitters for which no measurements are available [10,12]. Both of these approaches for REM prediction have their own challenges and benefits. For example, when planning for deployment of base stations (BS) in a geographical area, reactive REM prediction is less practical as there are no active transmitters from which sparse RSS measurements can be collected[1]. In contrast, proactive REM prediction is much more convenient for this problem as no RSS measurement is required. However, the lack of RSS measurements is also a significant challenge with proactive REM prediction. Specifically, what should be the basis for REM predictions? Most recent works in deep learning-based

[1]One way to deal with this limitation is wardriving with mobile BSs and mobile receivers; however, that is practically inconvenient.

proactive REM prediction have relied on training data collected from various geographical areas (not including the target areas where predictions will be made in the online/testing phase) via ray tracing. It is important to note that using ray tracing for collecting the training data is not an issue because the time needed to collect/generate training data does not affect the real-time operation in the online phase.

Goal of our work: Given the broader appeal of proactive REM prediction, in this paper, we consider the problem of predicting signal strength across an area due to a transmitter at a given location within the area. One of our primary motivations for investigating this problem was to address one of the *2023 IEEE ICASSP Signal Processing Grand Challenges*, namely, the *First Pathloss Radio Map Prediction Challenge* [13]. Unlike existing works in the literature, which are based on either relatively limited dataset [11], simulated maps [3], or 2D maps [10], the dataset considered in this challenge is a large-scale 3D dataset. This 3D dataset consists of over 701 geographical areas (henceforth city maps) with varying numbers of transmitters and building heights for each of the city maps. More details about this dataset can be found in [14] and are also described later in Section IV.

Our approach: To address the above problem, in this work, we develop an approach that relies on the u-net [15] neural network architecture. While u-net has been used in similar problems, we develop new strategies for adopting u-net for our problem with a 3D dataset. Additionally, we share several insightful findings that we discovered while participating in the radio map prediction challenge. First, we showed that using line-of-sight (LoS) information as an input to the neural network leads to better prediction accuracy. However, precomputing the LoS information incurs additional computation time, which hinders the primary goal of deep learning-based fast REM prediction. Hence, we developed three different methods for computing LoS information in our approach. These methods differ in terms of the quality of LoS information and computation time. Second, during our experiments, we learned that the density of buildings in the target area impacts the prediction accuracy. Hence, it is useful to train two different deep learning models: one based on city maps with low density of buildings and another based on city maps with high density of buildings. During the online phase, we can choose one of these two models based on the density of buildings in the target area. The impact of buildings' density has a higher impact, especially when the amount of training data is limited. Third, we identified that instead of training the neural network to predict the average signal strength, it might be helpful to train the neural network (actually two neural networks, as explained later) to predict the probability distribution of the signal strength. We showed that in certain scenarios, this approach can lead to better prediction accuracy. Additionally, as discussed in [3], predicting the distribution of signal strength can have additional benefits in specific applications.

Based on our learned lessons and insights, we used one particular combination of our developed strategies as our solution to the *ICASSP First Pathloss Radio Map Prediction Challenge*. Specifically, we used the buildings and transmitters location

and height information, along with LoS maps, stacked together as input to two u-nets to predict the probability distribution (mean and variance) of the signal strength. These two u-nets follow our third insight described above, in which we model the signal strength as a Gaussian random variable and train the two u-nets to predict the probability distribution (mean and variance) of the signal strength. The evaluation results, with unseen city maps, show that our proposed method provides an average normalized root-mean-square error (RMSE) of 0.045 with an average runtime of 14 milliseconds. Our approach and results for the *ICASSP First Pathloss Radio Map Prediction Challenge* are briefly summarized in [16]. However, in this paper, we share additional results that are not presented in [16].

In order to position our achieved REM prediction accuracy in the context of a relevant application, we consider the minimum propagation loss AP switch ON/OFF (ASO) strategy [17] proposed to address the excessive power consumption concerns in cell-free massive multiple-input multiple-output (CF-mMIMO) networks. The essence of the minimum propagation loss aware ASO (MPL-ASO) strategy is to activate only a subset of access points (APs) that is sufficient to meet user equipments (UEs) spectral efficiency (SE) based on the pathloss gain between APs and UEs [17] and set the rest of APs in sleep mode. In order to enable ASO strategies, existing works in the literature assume that all APs are frequently turned on to collect channel measurement, including LSF [17]–[19]. Alternatively, our CF-mMIMO use case demonstrates that predicted REMs of the off APs can be used in the AP selection problem of MPL-ASO, improving CF-mMIMO networks energy-efficiency by eliminating the need to frequently turn APs on. Our results show that by exploiting predicted REM, we attain an AP selection error of around 5% in case a UE's SE needs three extra APs.

A. Contributions

In summary, our contributions in this paper are the following:

- We present three different LoS calculation methods that rely only on a given transmitter location and the corresponding 3D city map, offering a tradeoff between accuracy and calculation time. In particular, these methods are per-pixel calculation, accelerated batch calculation, and neural-network-based calculation, all detailed in Section V. Unlike existing works in the literature, which only exploit binary LoS maps, our LoS maps present a fractional value for non-line-of-sight (NLoS) pixels, depending on the number of encountered buildings. This domain-knowledge-based information assists the neural network to learn in a swift and accurate manner. We quantify the performance gain of these three LoS calculation methods individually when utilized as a preprocessing step for REM prediction.
- We propose a u-net-based CNN to predict REM using transmitter location and city map information. We explore, in addition to LoS, also building density split and data augmentation as preprocessing steps. Furthermore, we present several insightful findings that we discovered

while participating in the radio map prediction challenge. In particular, these findings are 1) the performance gain when using LoS maps as an input to the neural network, 2) the positive impact of training two models based on the building density when the training dataset is limited, and 3) the performance gain obtained when training a u-net to predict the probability distribution instead of the average signal strength. We quantify these performance gains and impact using the *RadioMap3DSeer* dataset.

- We introduce a novel energy-efficient CF-mMIMO use case with REM-prediction-based MPL-ASO. We show that by relying on REMs predicted using our proposed u-net, along with data augmentation and LoS preprocessing steps, we eliminate the need to frequently turn sleep APs on to do channel measurements needed for MPL-ASO. We evaluate the performance of our proposed REM-prediction-based MPL-ASO, showing that it achieves an AP selection error of approximately 5% when using the predicted REM compared to the true one.

B. Organization

The rest of the paper is organized as follows. In Section II, we discuss the relevant related works. Next, in the Section III, we present a primer on u-net. Section IV presents our system model and describes the dataset used in this paper. We present our proposed methods in Section V, and the corresponding evaluation results in Section VI. Next, we present a use case of REM prediction, specifically, AP ON/OFF switching in CF-mMIMO, in Section VII. Finally, Section VIII provides the conclusions.

II. RELATED WORK

When sparse RSS measurements are available from the targeted active transmitter, i.e., for the case of reactive REM prediction, several methods have been investigated. The simplest method is to perform the predictions as a weighted average of the available measurements. For instance, in inverse distance weighting (IDW) methods, the weighting is done using a heuristic approach based on the inverse of the distance between the target location and the measurement location [2]. Another weighted-average-based example is the Kriging interpolation method, which uses a weighted average of the measured RSS, and obtains weights based on an optimization approach [20,21]. Alternatively, several works investigated the usage of deep learning for reactive REM prediction based on sparse measurements [3,22,23]. These works transformed the available information, e.g., sparse RSS measurements and their associated locations, transmitter locations, and environment map, into a set of stacked images (i.e., tensor) and fed it to a DNN to predict the REM. The deep neural networks used have an encoder-decoder structure, e.g., u-net [3], autoencoders [22], ResNets [23], so that the problem can be formulated as an image-to-image translation. In general, most of these works show the benefits of using deep learning over signal processing and heuristics methods, either in terms of REM prediction accuracy or prediction time.

On the other hand, when sparse measurements are unavailable for the targeted transmitter, i.e., for the case of proactive REM prediction, the most accurate REM prediction method is ray tracing [6]. However, as discussed in Section I, the computation time is the major drawback of ray tracing methods. The simplest way to perform the prediction is to use the free space radio wave propagation pathloss model that maps RSS to the distance from the transmitter [24]. However, this method does not take into account the shadow fading. Hence, various statistical modeling-based approaches have been developed, e.g., the log-normal shadowing model [25]. However, such radial-symmetric statistical methods, fail to capture the variation from one environment to another, e..g, location and height of buildings/obstacles.

An alternative promising way of performing proactive REM prediction is to collect training data via ray tracing from different cities/areas to learn a deep learning model that can perform predictions on unseen areas or transmitters [10,11]. In [11], the authors proposed a variant of the u-net architecture to predict the LSF maps of mmWave base stations. The authors used ray tracing to generate the training and ground-truth LSF maps from only three cities, considering terrain, transmitter location, buildings, foliage, as well as LoS information as input. A u-net-based method is introduced in [10] to predict the REM of unseen transmitters in a 2D scenario, with training data constructed from buildings maps and transmitters locations. In this work, the authors used maps of buildings layout from 6 cities, following the dataset presented in [14]. However, buildings, transmitters, and receivers are each set to constant heights across all training and testing data. The promising potential of u-net-based REM prediction and the substantial importance of filling the literature gaps regarding the lack of height information and the dataset size limitations inspire our work in this article.

III. PRIMER ON U-NET

In this section, we briefly present a primer on u-net which is the core of our proposed approach in this paper.

CNNs are arguably the most popular deep learning network architecture. CNNs are widely adopted in image processing tasks, demonstrating exceptional capabilities in image classification and object detection problems [26]. However, for pixel-wise image segmentation and regression, a different neural network architecture is required compared to those commonly used for image classification and object detection. For such problems, a special CNN architecture, known as u-net, has emerged as a promising solution [15].

The u-net architecture, introduced in [15], consists of an encoder contracting path, a decoder expansive path, and skip connections. In the *encoder path*, convolution and pooling operations are used to progressively reduce the spatial dimensions of the input while increasing the number of feature channels. This process helps in extracting hierarchical representations of the input image. The *decoder path* of the u-net is responsible for upsampling the feature maps back to the original input size. In this path, transpose convolutions (also known as deconvolutions or upsampling) are used to
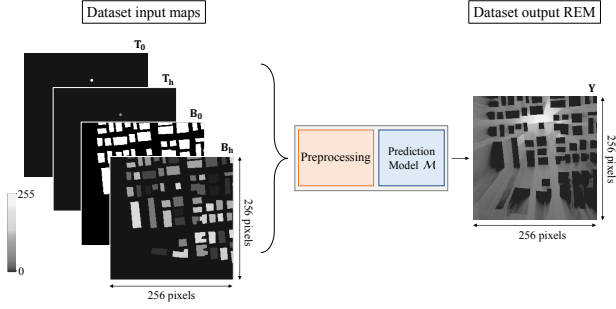
Fig. 1: A visualization of the considered REM prediction system, showing a sample input and the corresponding REM output. Our objective is to define data preprocessing steps and design a prediction model $\mathcal{M}$.

TABLE I: *RadioMap3DSeer* dataset Parameters.

Parameter	Value
Single map size	256×256 m^2
Simulation environment	Urban
Carrier frequency	3.5 GHz
System bandwidth	20 MHz
Transmit power	23 dBm
Antenna radiation pattern	Omnidirectional
Maximum path gain	-75 dB
Minimum truncated path gain	-111 dB
Buildings height range	6.6 - 19.8 m
Transmitter height	3 m above rooftop

progressively increase the spatial dimensions and reduce the number of feature channels. Since the upsampling in the decoder path lacks high-resolution information, *skip connections* are employed between the encoder and the decoder paths, copying and concatenating the feature maps in the encoder layers to the corresponding feature maps of the decoder layers. The skip connections help preserve spatial information and promote better segmentation. The resulting architecture is U-shaped, hence the name u-net.

IV. SYSTEM MODEL AND DATASET STRUCTURE

In this section, we introduce the system model, highlighting both the environment description and network assumptions. Subsequently, we present the details of the dataset used in this work.

A. System Model

The propagation environment considered in this work is an outdoor urban environment, with relatively narrow inter-building space and limited building heights ranging between 2 to 6 stories/floors. Transmitters, which are also known as base stations or APs, are placed on top of the buildings, whereas UEs are assumed to be on the ground at a constant 1.5 m height. Single omnidirectional antennas are assumed to be used by both transmitters and receivers, working at the center carrier frequency of 3.5 GHz with a system bandwidth of 20 MHz.

Problem Statement: Our main objective is to design a model $\mathcal{M}$, along with any companion preprocessing steps needed, taking 3D map information and transmitter locations as input and predicting the corresponding REM as an output, as depicted in Figure 1. In addition to the high prediction accuracy, we are also aiming at minimizing the prediction runtime of the model $\mathcal{M}$ to be in the scale of milliseconds.

B. Dataset Structure and Settings

In this work, we use the *RadioMap3DSeer* dataset [14], which consists of 701 city maps with buildings layouts fetched from *OpenStreetMap* [27] of 6 urban European-style cities, e.g., London, and Berlin. Intelligent ray tracing (IRT) [28] is used to simulate the REMs, using *WinProp* [29] software, assuming IRT with a maximum of two interactions (diffractions and/or reflections). Simulations were conducted for 80

different transmitter locations per map, resulting in a total of 56080 REMs. Each map is 256×256 m^2, stored as an image with a resolution of 256×256 pixels, implying that each pixel represents 1×1 m^2. In particular, as shown in Figure 1, dataset maps include:

- *Buildings layout*: Each of the 701 buildings layout maps is provided as a binary image as well as an image with quantized building heights. In the binary image, $\mathbf{B}_0$, pixels of buildings are set to ones and the rest to zeros, whereas in the quantized buildings height map, $\mathbf{B}_h$, pixels of buildings show the quantized height of buildings. Each building in a city is assigned a height ranging from 2 to 6 stories, with a constant story height of 3.3 m, resulting in buildings' heights ranging from 6.6 m to a maximum of 19.8 m. The buildings' heights were stored in $\mathbf{B}_h$ as uniformly quantized values between $[1, 255]$.

- *Transmitters locations*: Transmitters are assumed to be placed on buildings' rooftops, considering only buildings with a minimum height of 16.5 m located within the 150×150 center area to accommodate a transmitter. The transmitters are placed at the building edge with a height of 3 m from the corresponding building rooftop. Concerning each transmitter location, the dataset contains two 256×256 pixels images. The first image is a binary image denoted by $\mathbf{T}_0$, in which only the pixel containing the transmitter is set to one, and in the other image denoted by $\mathbf{T}_h$, only the pixel containing the transmitter presents the value of the corresponding building height.

- *Radio environment maps*: REMs are simulated using *WinProp* [29] software and path gain values of each 1×1 m^2 are stored in dB scale, considering a constant transmit power of 23 dBm. The maximum reported path gain is -75 dB and all path gain values below the analytical threshold of -111 dB are truncated, providing a path gain range of 36 dB $(-75 - (-111))$. Each REM, denoted by $\mathbf{Y}$, is scaled to gray levels between 0 and 255, enabling the authors [14] to save REMs as images.

Table I summarizes the main settings and parameters of the *RadioMap3DSeer* dataset [14] considered in this work.

V. AGILE RADIO MAP PREDICTION

In this section, we present our proposed approach. First, we present the architecture of u-net that we use in all of our methods unless otherwise stated. Then, we describe the
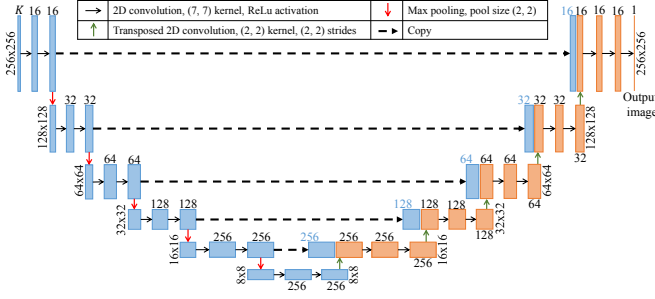
Fig. 2: A detailed architecture of the U-net used in this work.

Fig. 3: A visualization of the proposed U-net model assisted by providing $\mathbf{L}_f$ as an additional input, denoted by $\mathcal{M}(\text{noDAug}, \text{LoS}_f, \text{U-net}, \text{MSE})$.

different possible preprocessing steps that we investigate in this paper. Finally, we present an alternative neural network architecture that we used as part of the ICASSP challenge.

A. U-net architecture

As discussed in Section III, u-net architecture can achieve superior segmentation performance. In the context of our problem, the REM prediction is similar to segmentation, with the only difference being that for each of the output pixels, we need to perform regression instead of classification. This idea has been explored in some prior works on REM prediction and serves as our motivation for using u-net. The u-net architecture uses an image-to-image translation approach. Hence, the available information for making the REM predictions must be transformed into images. Conveniently, the available information is encoded as images in the dataset described in Section IV (refer to $\mathbf{B}_0$, $\mathbf{B}_h$, $\mathbf{T}_0$, and $\mathbf{T}_h$). We stack several images as a 3D tensor and feed it to the u-net. Accordingly, the operation of the u-net can be formally described as the following mapping $\mathcal{M} : \mathcal{R}^{256 \times 256 \times K} \rightarrow \mathcal{R}^{256 \times 256}$. The value of K depends on the number of images we use as input. The architecture of u-net that we primarily rely on is shown in Figure 2. We train $\mathcal{M}$ by minimizing the mean squared error between the true REM $\mathbf{Y}$ and the estimated REM $\hat{\mathbf{Y}}$.

B. Using LoS information as additional input to u-net

In addition to using the available information from the dataset as input to our neural network, we use an additional input, which we call the LoS map, $\mathbf{L}_f$. Specifically, in this section, we use $K = 3$ and the three input images to the u-net are $\mathbf{T}_h$, $\mathbf{B}_h$, and $\mathbf{L}_f$. As discussed next, we compute $\mathbf{L}_f$ using $\mathbf{T}_h$ and $\mathbf{B}_h$. Since both $\mathbf{T}_h$ and $\mathbf{B}_h$ are used as input to the neural network, the neural network should be able to learn and perform equally well with and without using $\mathbf{L}_f$ as an additional input. However, the learning task of the neural network without $\mathbf{L}_f$ would be more complex. Hence, based on our domain knowledge of radio wave propagation, we assist the neural network in learning quickly by providing $\mathbf{L}_f$ as an additional input, as illustrated in Figure 3. We show later in Section VI that, indeed, precomputing the LoS map and using it as an input to the neural network improves the REM prediction accuracy. In the following, we describe three different methods for computing $\mathbf{L}_f$.
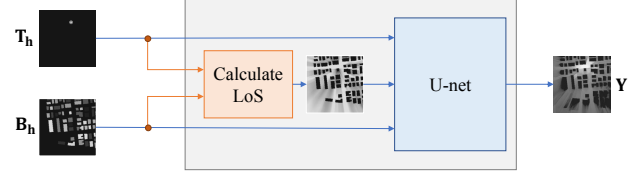
1) Per-pixel LoS calculation (PxLoS): For a given transmitter location, (x_t, y_t, z_t), we compute the LoS information for each of the pixels in the area where the transmitter is located. First, for a particular pixel, say (x_r, y_r), we form a straight line, l_{tr}, in the 3D space between (x_t, y_t, z_t) and (x_r, y_r, z_r). Note that (x_r, y_r) denote the pixel center in the XY plane and $z_r = 1.5$ m as receivers are assumed to be always 1.5 m above the ground level as discussed in Section IV. The straight line l_{tr} is defined as an ordered sequence of N_l coordinates $\langle (x_l, y_l, z_l) \rangle; l = 1, 2, ..., N_l$, where N_l is the number of pixels that l_{tr} passes through. The set of pixels, $\mathcal{P} = \{(x_l, y_l)\}; l = 1, 2, ..., N_l$, associated with l_{tr} are found using the Bresenham's line algorithm [30]. In this method, we move from x_r to x_t one pixel at a time and find the pixels along y dimension based on the slope of the line such that the sequence of pixels in $\mathcal{P}$ closely approximates a straight line between (x_r, y_r) and (x_t, y_t). For computing the values of z_l, we compute the slope along z dimension as $\frac{z_t - z_r}{x_t - x_r}$ and use $z_l = z_r + (x_l - x_r) \times \frac{z_t - z_r}{x_t - x_r}$. Then, for each of the pixels in $\mathcal{P}$, we check the building height for that pixel from $\mathbf{B}_h$. For a particular pixel in $\mathcal{P}$, if the building height is less than z_l (attribute of l_{tr}), we say that the pixel is in LoS. Based on this checking, we form another set, $\mathcal{L}$, that has the same size as $\mathcal{P}$ and has a one-to-one association with the pixels in $\mathcal{P}$. The elements of $\mathcal{L}$ are either 0 or 1, depending on whether the corresponding pixel is in LoS or not. Finally, for $\mathbf{L}_f$, we set the value of pixel (x_r, y_r) to be $\left(1 - \frac{\sum_{i \in \mathcal{L}} \mathbf{1}_i}{|\mathcal{L}|}\right)$ where $\mathbf{1}_i$ is an indicator function. $\mathbf{1}_i = 1$ if the ith element of $\mathcal{L}$ is 1, and 0 otherwise.

The primary disadvantage of the PxLoS method is that it is very slow as it performs the LoS check pixel by pixel for each transmitter. Specifically, if the target area has $P \times P$ pixels, then the complexity of generating or is $O(P^3)$.

2) Accelerated batch LoS calculation (AbLoS): The primary reason for the slowness of PxLoS is that the elements of $\mathbf{L}_f$ are computed sequentially. To avoid that, in AbLoS, we simultaneously compute all the elements of $\mathbf{L}_f$. Computing $\mathbf{L}_f$ in one shot is especially attractive because we can use libraries that can vectorize the operations in AbLoS and accelerate the computation of $\mathbf{L}_f$ in GPU. The details of this approach are described next.

This approach is also based on Bresenham's algorithm. The basic idea of Bresenham's algorithm is to find the set of n-D pixels that closely approximate a n-D line between two points. In our problem, n is 3. For a given transmitter location, (x_t, y_t, z_t), first, let us denote the set/batch of target 3D pixels for which LoS information must be computed as $\mathcal{B}$. Each of the elements of $\mathcal{B}$ is defined by (x_r, y_r, z_r), where (x_r, y_r)

denote the pixel center in the XY plane and z_r is always 0. Using $z_r = 0$ instead of 1.5 m is an approximation. Next, based on Bresenham's algorithm, we move from (x_r, y_r, z_r) to (x_t, y_t, z_t) one pixel at a time along the driving dimension. Here, driving dimension is the dimension among x, y, and z that has maximum absolute difference between (x_r, y_r, z_r) and (x_t, y_t, z_t), i.e., $\max\{|x_t - x_r|, |y_t - y_r|, |z_t - z_r|\}$. For example, if the driving dimension is x, then we traverse $x_r, x_r + 1, ..., x_t$. While we move along the driving dimension, we also keep updating the other two dimensions, using pixel-wise increments based on the slope of the line between (x_r, y_r, z_r) and (x_t, y_t, z_t). For example, if the driving dimension is x, we define the slope along y dimension as $\frac{y_t - y_r}{|x_t - x_r|}$ and the slope along z dimension as $\frac{z_t - z_r}{|x_t - x_r|}$. Although the slope can be fractional, the increment along non-driving dimensions is always in integers. This is done by rounding off fractional values to the nearest integers. This procedure gives us a set of 3D pixels that closely approximate a line between (x_r, y_r, z_r) and (x_t, y_t, z_t). Importantly, in AbLoS, we assume that the distance to (x_t, y_t, z_t) along the driving dimension from all the pixels in $\mathcal{B}$ are the same. Specifically, we assume this distance, say D, to be the maximum absolute difference between any pair (x_r, y_r, z_r) and (x_t, y_t, z_t), along any dimension (x, y, z). I.e., $D = \max_B \max\{|x_t - x_r|, |y_t - y_r|, |z_t - z_r|\}$. This allows us to find the approximate lines, as explained before, between all the elements of $\mathcal{B}$ and (x_t, y_t, z_t) simultaneously. Consequently, this vectorized operation can be significantly accelerated on a GPU using the software library CuPy [31]. Next, for each of the computed lines, we check the number of constituent 3D pixels that intersect with the buildings. This can be done by converting $\mathbf{B}_h$ into a set of 3D pixels filled with 1s and 0s depending on whether a pixel is under a building or not and finding its intersection with the computed lines. Let us denote p_r as the number of such intersecting 3D pixels for the line between (x_r, y_r, z_r) and (x_t, y_t, z_t). This operation can also be accelerated as all the lines comprise the same number of 3D pixels. Finally, for $\mathbf{L}_f$, we set the value of pixel (x_r, y_r) to be $\left(1 - \frac{p_r}{D}\right)$.

AbLoS uses an approximation that all the lines are made up of the same number of 3D pixels. However, it is much faster than PxLoS. In the above description, we have mentioned a couple of crucial operations where GPU-based acceleration is useful. Additionally, in our implementation, we leverage CuPy to accelerate other operations whenever possible.

3) Predicting LoS map via neural network (NNLoS): In this method, which we call NNLoS, we predict the LoS map instead of computing it. Specifically, we train a neural network that can approximate the LoS maps generated by the PxLoS method.

During the training phase, for each of the training examples in the dataset, first, we compute $\mathbf{L}_f$ using the PxLoS method. These LoS maps become the labels for training the neural network in NNLoS. Recall from Section V-B1 that for computing $\mathbf{L}_f$ we need $\mathbf{T}_h$ and $\mathbf{B}_h$. Hence, we use $\mathbf{T}_h$ and $\mathbf{B}_h$ stacked as a 3D tensor as input to the neural network and train it to predict $\mathbf{L}_f$. Let us denote this LoS predictor neural network as a function, $\mathcal{F}_{LoS}$, that performs the following mapping
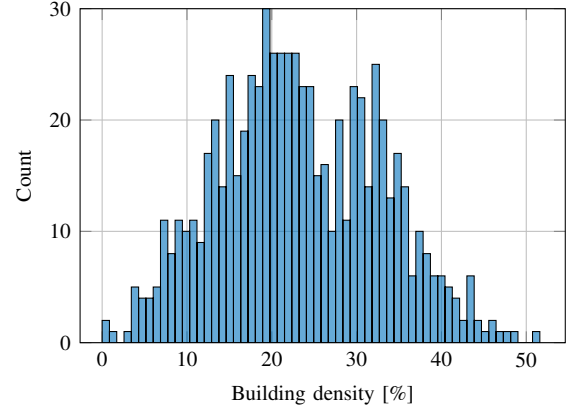

Fig. 4: A histogram of the building density (no. of buildings pixels divided by the total number of pixels).

$\mathcal{F}_{LoS} : (\mathbf{T}_h | \mathbf{B}_h) \rightarrow \mathbf{L}_f$, where $|$ denotes the depth-wise stacking. Since $\mathcal{F}_{LoS}$ performs an image-to-image translation, we use a u-net to represent this function too. For the $\mathcal{F}_{LoS}$ u-net, we use the same neural network architecture as that for signal strength prediction (refer to Figure 2). We train $\mathcal{F}_{LoS}$ by minimizing the mean squared error between $\mathbf{L}_f$ and its estimation, $\hat{\mathbf{L}}_f$ using the Adam optimizer with learning rate of 10^{-4}.

During prediction, we simply use the trained neural network, i.e., $\mathcal{F}_{LoS}$ to predict the LoS map $\hat{\mathbf{L}}_f$. The computation time of $\hat{\mathbf{L}}_f$ is, in general, much lower than computing the actual LoS map, $\mathbf{L}_f$, via PxLoS. However, since $\hat{\mathbf{L}}_f$ is an estimation of $\mathbf{L}_f$, it negatively affects the REM prediction, as shown later in Section VI.

C. Environment building density

Radio channels are commonly modeled based on the characteristics of the propagation environment by either using different models for different environments, or a single model with different environment-dependent parameter values. For instance, in the log-normal pathloss model, the pathloss exponent has different values for environments such as urban, suburban, indoor, etc. [25]. A key element that characterizes outdoor propagation environments is the building density, which influences both small-scale multipath fading as well as large-scale shadow fading. Intuitively, the buildings' density noticeably varies when comparing dense-urban, urban, or suburban environments. Inspired by the influence buildings' density has on radio wave propagation, in this section, we explore the building density in the 701 maps provided in the *RadioMap3DSeer* dataset. We define building density in a given city map as the number of buildings pixels divided by the total number of pixels. Figure 4 presents the histogram of the buildings' density per map. As shown in the figure, a bimodal distribution with two peaks can be spotted around 20% and around 31%. These two peaks imply that there are two different groups of building density, which can be translated to two different propagation environments, e.g., dense-urban and urban. Training a deep learning model using a dataset that depicts a bimodal distribution could potentially
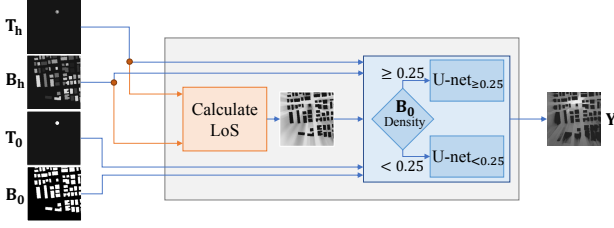
Fig. 5: A visualization of the models used, where two U-nets are used during training and prediction each for a different building density. In case $PxLoS_f$ is used the two models are $\mathcal{M}(\text{noDAug}, PxLoS_f, \text{U-net}_{\geq 0.25}, \text{MSE})$, and $\mathcal{M}(\text{noDAug}, PxLoS_f, \text{U-net}_{<0.25}, \text{MSE})$
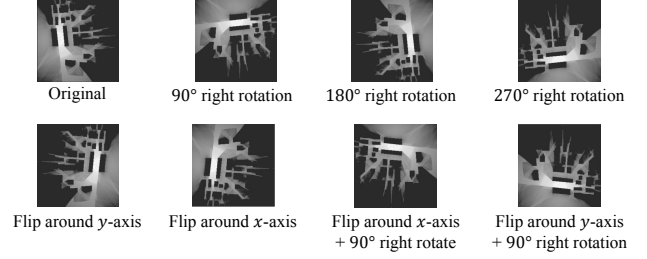

Fig. 6: The considered data augmentation technique performed on a sample REM image, which in addition to the original image results in $\times 8$ nonidentical images.
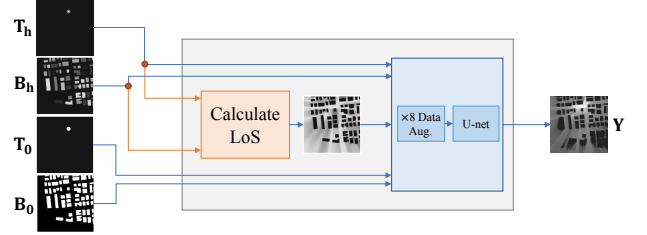
cause an underfitting problem. While the large-scale size of the dataset considered in this work should be sufficient to avoid underfitting problems, the tradeoff is the excessive training time needed to train a model using the whole dataset.

An alternative approach to address the underfitting problem, especially when using a limited dataset size, is to follow the same intuition used in modeling propagation environments and design a *specialized* model for the targeted environment. To this end, we select a building density threshold of 25% to split the dataset and train two different models. Using a building density threshold of 25% results in 418 maps with building density > 25% ($\approx$ 60% of the dataset) and 282 maps with building density $\leq$ 25% ($\approx$ 40%). In order to train two different models based on the building density, for a given set of training maps, namely, $\mathbf{B}_0$, $\mathbf{B}_h$, $\mathbf{T}_0$, $\mathbf{T}_h$, and $\mathbf{L}_f$, with the corresponding output image $\mathbf{Y}$, we first check the building density in $\mathbf{B}_0$. As depicted in Figure 5, if it is $\geq$ 25% we use the corresponding training sample in training a u-net model, denoted by U-net$_{\geq 25}$, otherwise we use the training sample in training a u-net model denoted by U-net$_{<25}$. Similarly, in the test phase, we first check the building density in $\mathbf{B}_0$, and subsequently use the corresponding u-net model for the REM prediction. The performance evaluation of REM prediction when using two buildings-density-dependent models follows in Section VI.

D. Data augmentation

A common practice when building deep learning models is data augmentation employed to increase the model's ability to generalize as it adds variability to the data, which in turn minimizes overfitting. This practice is particularly advantageous in REM prediction problems as it saves on the time and cost of collecting, or using ray tracing to simulate, additional labeled data. Despite the large-scale dataset considered in this work, data augmentation showed performance gains when using horizontal, vertical, and diagonal flips [32], resulting in, including original images, $\times 4$ dataset size. In order to take advantage of data augmentation benefits, while maintaining a consistent 256$\times$256 pixels image size, we adopt a data argumentation technique that covers all possible rotations and flips, resulting in 8 nonidentical versions of an image, and hence a $\times 8$ dataset size. In particular, for our data augmentation technique, we consider the rotations and flips illustrated in Figure 6. The data argumentation technique is applied during the training phase on all input images, i.e., $\mathbf{B}_0$, $\mathbf{B}_h$, $\mathbf{T}_0$, $\mathbf{T}_h$, and $\mathbf{L}_f$, as

Fig. 7: A visualization of the model used with a $\times 8$ data augmentation step, denoted by $\mathcal{M}(\text{DAug}, PxLoS_f, \text{U-net}, \text{MSE})$.

well as on the corresponding output image $\mathbf{Y}$, as visualized in Figure 7. We present the performance gains obtained by performing such a data argumentation technique on the dataset in the CF-mMIMO use case in Section VII. In this use case, data argumentation plays a crucial role due to the limited size of the considered dataset.

Remark. In scenarios where we examine building-density-based data split and data augmentation, we deal with rather limited dataset size, e.g., only 100 city maps or even one in the CF-mMIMO use case (cf. Section VII). In these scenarios, we opt for u-net input supported by $\mathbf{B}_0$ and $\mathbf{T}_0$, as these two extra images enrich the input data and positively affect the RMSE performance. This positive impact disappears in cases where we significantly increase the size of the training data, e.g., 600 maps, and ends up being a burden that slows the training time. Therefore, in such scenarios, we only use $\mathbf{B}_h$, $\mathbf{T}_h$, and $\mathbf{L}_f$ to construct the input.

E. Alternative deep learning architecture based on Kullback–Leibler (KL) divergence loss

In this section, we present an alternative deep learning architecture for REM prediction. This architecture can be used with all the preprocessing discussed previously. In this approach, instead of using the mean-square error (MSE) loss function, we exploit the KL divergence loss function [33]. Specifically, we assume that the probability distribution of signal strength follows the Gaussian distribution and train two identical u-nets, one for mean and one for variance prediction, for estimating the distribution of the signal strength, as depicted in Figure 8. We use the KL divergence as the loss function as it is well suited for measuring differences in probability distributions. Our primary reason for using this loss function is that it is known to act as an intelligent regression function: locations for
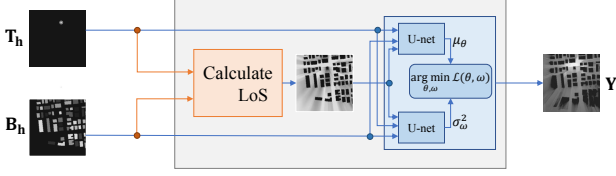
Fig. 8: A visualization of the KL-based model architecture adopted in [3], in which two identical u-nets are used, one for mean and one for variance prediction.

which the model learned to predict high uncertainty will have a smaller effect on the loss. Such locations are often the ones with extremely low gain, where prediction is challenging for deep-learning models. After training, the u-net for estimating variance is discarded and the one for estimating mean is used for REM prediction in the test phase. We have used this approach for REM prediction, but for a different problem setup, in one of our previous works [3]. For the sake of brevity, we do not repeat the details here, but interested readers are encouraged to refer to [3] for more details. The architecture of the u-nets used in this approach is the same as the one used in [3], with the only difference being the size of the radio maps: 256×256 in this paper and 64×64 in [3].

U-net models notation: Considering the different LoS pre-processing methods, data augmentation, and environment building density, we use $\mathcal{M}(a, b, c, d)$ to denote the different u-net-based models we explore in this work, where

- $a \in \{\text{DAug, noDAug}\}$ corresponds to data augmentation and no data augmentation.
- $b \in \{\text{noLoS}, \text{PxLoS}_f, \text{AbLoS}_f, \text{NNLoS}_f\}$ indicates the type of LoS calculation used.
- $c \in \{\text{U-net}, \text{U-net}_{\geq 25}, \text{U-net}_{<25}\}$
- $d \in \{\text{MSE, KL}\}$

VI. EVALUATION RESULTS OF REM PREDICTION

This section details the evaluation results of our REM prediction approach, quantifying the impact of our key design parameters, namely, LoS maps, building-density-based training, and the loss function. In this section, we used *RadioMap3DSeer* dataset maps 0-500, 500-600, and 600-700 for model training, validation, and testing, respectively, unless otherwise mentioned. At the end of this section, we also present the results we submitted to the ICASSP challenge test set.

A. Performance Metric

Considering the output REM $\mathbf{Y}$ and the corresponding estimated REM $\hat{\mathbf{Y}}$, we define the RMSE of a test set containing L maps as

$$\mathcal{E}_{\text{RMSE}} = \sqrt{\frac{\sum_L \sum_N |\mathbf{Y} - \hat{\mathbf{Y}}|^2}{LN}}, \quad (1)$$

where N is the total number of pixels, which equals to 256×256. Note that this RMSE slightly differs from the RMSE used in the *2023 IEEE ICASSP First Pathloss Radio Map*
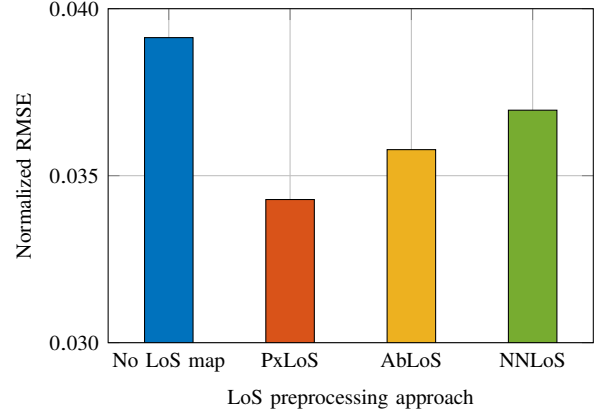

Fig. 9: Impact of different LoS preprocessing approaches on REM prediction accuracy.

Prediction Challenge [16,32], where building locations were set to zero before calculating the RMSE of a given map. By setting buildings locations to zero, one corrects any estimation error in the building pixels of the estimated REM, slightly underestimating the RMSE error compared to the RMSE in (1).

B. Impact of LoS maps

In Figure 9, we use the u-net architecture of Figure 2 for REM prediction. For the bar labeled 'No LoS map,' no LoS map is used, and only $\mathbf{T}_h$ and $\mathbf{B}_h$ are used as u-net input, i.e., $K = 2$. For the remaining three bars, we use $\mathbf{T}_h$, $\mathbf{B}_h$, and $\mathbf{L}_f$ as u-net input, i.e., $K = 3$. The labels on X-axis in Figure 9 indicate the method for computing the LoS map, $\mathbf{L}_f$. We make several observations for this figure.

First, by comparing the REM prediction accuracy for the 'No LoS map' based u-net and PxLoS-based u-net, we see that including the LoS map significantly improves the prediction accuracy; specifically, the prediction accuracy improves by 13%. As discussed in Section V-B, this happens because using the LoS map, based on domain knowledge of RF propagation, helps the u-net to learn a better mapping easily.

Second, the AbLoS-based u-net and NNLoS-based u-net also perform better than the 'No LoS map' based u-net. This is expected due to using LoS maps in AbLoS-based u-net and NNLoS-based u-net. However, we observe that the REM prediction accuracy with AbLoS-based u-net and NNLoS-based u-net is lower than that with PxLoS-based u-net. The reason is that both AbLoS and NNLoS use more approximations than PxLoS when computing the LoS map. As explained in Section V-B, NNLoS tries to mimic PxLoS via a neural network, and AbLoS assumes all the pixels are equidistant from the transmitter along the driving dimension.

Third, we see that AbLoS-based u-net performs slightly better than NNLoS-based u-net. To better understand the reason behind that, we also plot the generated LoS maps using three methods for one scenario in Figure 10. This figure shows that the LoS map predicted by NNLoS is very similar to the one computed by the PxLoS method, and the LoS map created by AbLoS looks more different than the other two. Hence,

TABLE II: Average prediction time per REM. This includes both preprocessing and forward pass of the neural network.

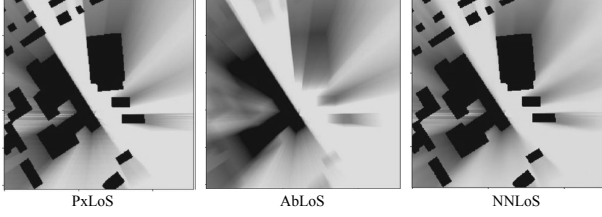| No LoS map	PxLoS	AbLoS	NNLoS
6.5 msec	3×10^3 msec	14 msec	13 msec

Fig. 10: Visual comparison of LoS maps generated by different methods.

we expect NNLoS-based u-net to have better accuracy than AbLoS-based u-net. However, that is not the case in Figure 9. We believe the reason for this discrepancy is the following.

In NNLoS-based u-net, first, we train the LoS predictor u-net using all the available training examples. Then when the REM predictor u-net is trained, we use the LoS maps predicted by NNLoS for all the training examples. I.e., the predicted LoS maps are for the same set of examples that were used for the training of NNLoS. This way, the REM predictor u-net is trained on LoS maps that are biased. This limits the generalization capability of the NNLoS-based u-net and adversely impacts the REM prediction accuracy on unseen test data.

Next, we compare the different methods for LoS map generation in terms of overall prediction time in Table. II. Here, overall prediction time is the sum of the time required for generating the LoS map and predicting the REM. Although PxLoS-based u-net has the highest REM prediction accuracy, we see from Table. II that its overall prediction time is the maximum. Both AbLoS-based u-net and NNLoS-based u-net are significantly faster than PxLoS-based u-net, but their overall prediction time is almost double the prediction time of the 'No LoS map' based u-net. 'No LoS map' based u-net has the lowest overall prediction time because it performs no preprocessing, only forward pass of the u-net. Lastly, we see that both AbLoS-based u-net and NNLoS-based u-net have comparable overall prediction times.

C. Building density

Going beyond the challenge-required results, we also consider scenarios with constraints on the size of training data and training time. Despite the nearly instant prediction time of our proposed model, which takes only a few milliseconds per map, the training time with 600 $\times$ 80 maps the whole dataset takes tens of hours, depending on the available computational resource, i.e., the number of GPU clusters. In Figure 11, we present the normalized RMSE performance when training two different u-net models, $\mathcal{M}(\text{noDAug}, \text{PxLoS}_f, \text{U-net}_{\geq 25}, \text{MSE})$ and $\mathcal{M}(\text{noDAug}, \text{PxLoS}_f, \text{U-net}_{<25}, \text{MSE})$, based on the map's building density, compared to the case in which a single u-net $\mathcal{M}(\text{noDAug}, \text{PxLoS}_f, \text{U-net})$ is used. This figure investigates

TABLE III: Summary of the results we presented in the challenge using a new unpublished test set from the organizers.

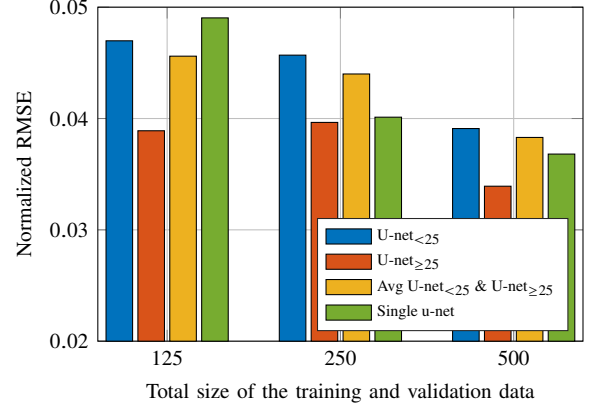| Model	Norm. RMSE	Run Time [ms]
$\mathcal{M}(\text{noDAug}, \text{noLoS}, \text{U-net}, \text{MSE})$	0.051	6
$\mathcal{M}(\text{noDAug}, \text{AbLoS}_f, \text{U-net}, \text{MSE})$	0.046	14
$\mathcal{M}(\text{noDAug}, \text{AbLoS}_f, \text{U-net}, \text{KL})$	0.045	14

Fig. 11: Impact of using two separate models based on the building density compared to the case of using a single model. Training-validation split is 80-20 %.

scenarios with constraints on the training time or the size of the training data due to limitations on training computational time or resources. The figure considers three sizes of training data, 125, 250, and 500, all split 80% for training and 20% for validation. In all the models shown in Figure 11, maps 600-700 from the *RadioMap3DSeer* are used as a test set. In the case of $\mathcal{M}(\text{noDAug}, \text{PxLoS}_f, \text{U-net}_{\geq 25}, \text{MSE})$ and $\mathcal{M}(\text{noDAug}, \text{PxLoS}_f, \text{U-net}_{<25}, \text{MSE})$ models, the dataset is approximately 40% and 60%, respectively, both for training and validation. However, the test set is biased with 84% used the model $\mathcal{M}(\text{noDAug}, \text{PxLoS}_f, \text{U-net}_{<25}, \text{MSE})$ and 16% used the model $\mathcal{M}(\text{noDAug}, \text{PxLoS}_f, \text{U-net}_{\geq 25}, \text{MSE})$. Note that we used this test set despite its bias to ensure consistency across the paper. The training and test process follows the block diagram shown in Figure 5. As shown in the figure, in the case of a limited dataset, e.g., 125 maps, training a model based on the building density in the area of interest outperforms the case in which a single model is used without paying attenuation to the building density. However, increasing the size of the training data gives an edge performance of 0.004 in the case where a single model is used compared to the case where building-density-based models are used. This difference in performance diminishes to only 0.001 in the case where the size of the training dataset is 500, indicating that all models considered in the figure have sufficient data to generalize and avoid overfitting.

D. Impact of KL divergence loss

In Figure 12, we show the impact of the loss function on REM prediction. Specifically, for the bar labeled 'MSE,' we used the u-net of Figure 2 and trained it using the mean squared error (MSE) loss. For the bar labeled 'KL,' we used the u-net architecture and loss function as discussed in
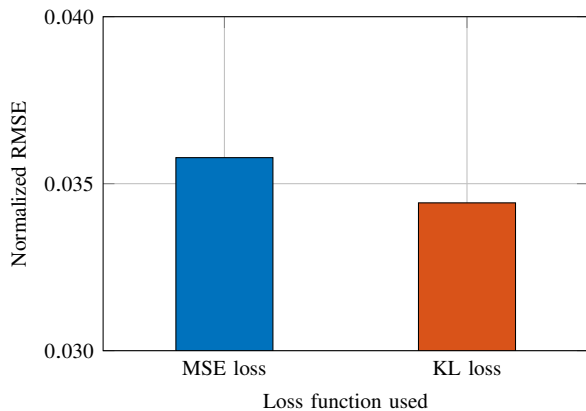
Fig. 12: Impact of loss function on the REM prediction.

Fig. 13: A representation of a CFmMIMO network with ASO.

Section V-E. For both of these approaches, we use $\mathbf{T}_h$, $\mathbf{B}_h$, and $\mathbf{L}_f$ as input, and $\mathbf{L}_f$ is computed using the AbLoS method. We observe from this figure that using the KL divergence loss provides some improvement in REM prediction accuracy over the MSE loss.

E. Challenge Results

The proposed models are evaluated on a new unpublished test set from the challenge organizers, consisting of 84 city maps, each with 80 different transmitter locations, resulting in a total of 6720 radio maps. Similar to the *RadioMap3DSeer* dataset, receivers are assumed to have a fixed height of 1.5 m, and transmitters are assumed to be located 3 m above buildings' rooftops. Table III summarizes the results obtained using our proposed models. As shown in the table, $\mathcal{M}(\text{noDAug}, \text{AbLoS}_f, \text{U-net}, \text{KL})$ outperforms $\mathcal{M}(\text{noDAug}, \text{AbLoS}_f, \text{U-net}, \text{MSE})$ and $\mathcal{M}(\text{noDAug}, \text{noLoS}, \text{U-net}, \text{MSE})$, with 8 ms extra runtime for LoS calculation as a tradeoff compared to $\mathcal{M}(\text{noDAug}, \text{noLoS}, \text{U-net}, \text{MSE})$.

VII. USE-CASE: AP ON/OFF SWITCHING IN CF-MMIMO

CF-mMIMO is a promising novel wireless network architecture proposed to address the inadequate cell-edge users performance, which might experience tens of dB weaker channel compared to cell-center users [34]. Instead of using a single AP to serve a UE, in CF-mMIMO networks, a UE is alternatively served by several, or even all, APs [35]. In CF-mMIMO networks, distributed APs jointly operate to coherently serve UEs on the same time/frequency resource, using spatial multiplexing. This joint operation of distributed APs ensures not only a higher signal-to-noise ratio (SNR), but also better multi-user interference suppression when compared to the conventional cellular networks [34]. However, in order to harness the advantages of CF-mMIMO networks, a dense deployment with large number of APs is needed. While the APs used in CF-mMIMO are expected to be significantly less complex than conventional base stations used in cellular networks, the large-scale deployment needed raises concerns about the overall power consumption and the corresponding energy-related pollution [36].
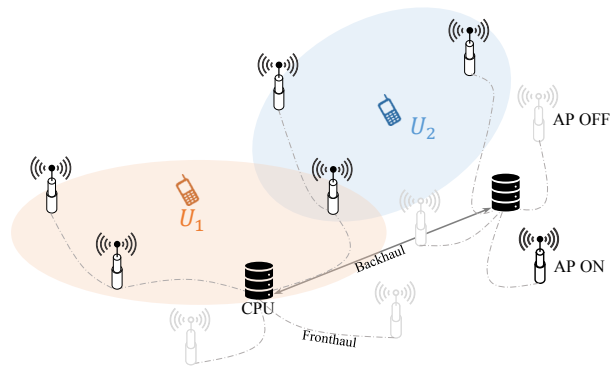
In order to address the power-consumption concerns in CF-mMIMO networks, strategies known as ASO are attracting considerable research focus [36]. ASO strategies suggest that APs should be dynamically switched on (put in active mode) and off (put in sleep mode) based on the UEs traffic demand. In particular, ASO strategies consider the AP status as an optimization variable, aiming at selecting a subset of APs that meets UEs SE requirements while the remaining APs are switched off. Selecting the optimal energy-efficient subset of APs that meets a given single or multiple UEs SE requirements is an NP-hard problem, requiring assessing all possible combinations of APs [19]. The authors in [19] presented a global solution of the AP selection problem by solving a computationally-intensive non-convex optimization problem. To address the computational complexity of the non-convex optimization problem, heuristic ASO strategies based on the location and propagation losses between APs and UEs are introduced in [17]–[19], including ASO strategies such as random selection ASO (RS-ASO), nearest neighbor ASO, Chi-square test-based ASO (ChiS-ASO), optimal energy-efficiency-based greedy ASO, and MPL-ASO. Among these strategies, MPL-ASO reportedly provided a good trade-off between the SE performance and complexity, exploiting large-scale fading coefficients between APs and UEs with a minor performance penalty [17,36].

ASO strategies that depend on propagation losses assume that APs are frequently turned on to collect the measurements needed to select the optimal set of APs, which implies that some APs might wake-up to do channel measurements and end up not serving any UEs, negatively affecting the energy efficiency of the whole network. This particular problem, combined with the promising performance of MPL-ASO, motivates our work on the ASOs use-case. In this section, we present a use-case demonstrating that one can exploit the swift and accurate REM prediction to boost the energy efficiency of MPL-ASO by eliminating the need to unnecessarily turn on APs to do channel measurements. By using the subset of active APs, we train our u-net-based model to predict the large-scale fading of the off APs, and subsequently use the predicted large-scale fading to decide on which extra APs to switch on. In the following, we detail the system model and the evaluation results.

Algorithm 1 LSF Prediction-based MPL-ASO

1: **Input:** $\mathbf{B}_h$, $\mathbf{B}_0$, $\mathbf{T}_h$, $\mathbf{T}_0$, $\mathcal{A}_a$, UE's location and SE demand.

2: **Output:** $\hat{\mathcal{A}}_{sa}$

3: Train prediction model $\mathcal{M}$ using $\mathcal{A}_a$
4: **for** each extra SE demanding UE **do**
5:    $\hat{\mathbf{Y}}_u \leftarrow \mathcal{M}(\text{DAug}, \text{PxLoS}_f, \text{U-net}, \text{MSE}) \; \forall \, \mathcal{A}_s$
6:    Order APs in $\mathcal{A}_s$ based on $\hat{\mathbf{Y}}_u$
7:    $\hat{\mathcal{A}}_{sa} \leftarrow$ Orderly activate APs from $\mathcal{A}_s$ until SE is satisfied
8: **end for**
9: **return** $\hat{\mathcal{A}}_{sa}$

A. System Model

We consider a CF-mMIMO network with M single-antenna APs deployed in an outdoor environment, serving N_u arbitrarily distributed users in an outdoor area of interest. APs are connected via the so-called fronthaul connections to one or multiple edge-cloud central processing units (CPUs), which are linked via an optical fiber or a microwave backhaul link, as depicted in Figure 13. We assume that an MPL-ASO strategy is implemented in the network, which implies that each of the APs can be either in active/ON mode or sleep/OFF mode, depending on LSF fading between the APs and UEs. At any given time instant, we have a set of active APs, $\mathcal{A}_a = \{A_1, \ldots, A_{M_a}\}$ and a set of sleep APs, $\mathcal{A}_s = \{A_1, \ldots, A_{M_s}\}$, with $M_a + M_s = M$, and $M_a \neq 0$. In case a connected UE demands extra downlink SE, the CPU coordinates the MPL-ASO strategy, deciding on the number and the addresses of APs to be switched on. It is worth noting that in this use-case, we mainly focus on the performance of MPL-ASO with predicted LSF path gains; for a detailed formulation of the downlink SE, we refer the reader to Chapter 6 in [34].

B. LSF-Prediction-Based MPL-ASO

Our main objective is to exploit LSF to decide on which extra APs to switch on in order to join the already active APs in serving the extra SE demanding UE. In other words, we aim at finding a subset of sleep APs to reactivate, $\mathcal{A}_{sa} = \{A_1, \ldots, A_{M_{sa}}\} \subseteq \mathcal{A}_s$, with $M_{sa} \leq M_s$. To this end, we train a model $\mathcal{M}$ using the LSF information, which is assumed to be collected from the set of active APs $\mathcal{A}_a$ and outdoor users they served throughout their active time window. We explore the case in which the full REM of active APs is available, as well as the case where only LSF information from a set of randomly distributed users is available. Using the obtained $\mathcal{M}$ model, we stack the predictions of all APs $\in \mathcal{A}_s$ in $\hat{\mathbf{Y}}_u$, and subsequently, order them based on LSF to the UE of interest from lowest to highest. We orderly activate APs from the set $\mathcal{A}_s$, i.e., add them to the estimated subset $\hat{\mathcal{A}}_{sa}$ until the UE's SE is satisfied. We summarized the steps of the LSF-prediction-based MPL-ASO in Algorithm 1. In the following, we present the evaluation results concerning our LSF-prediction-based MPL-ASO use case.
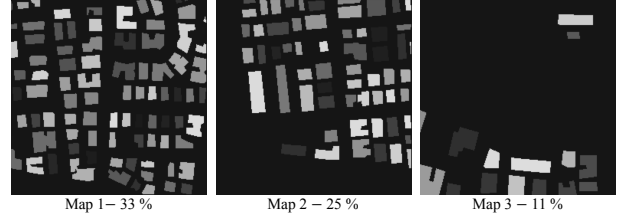

Fig. 14: Maps layout considered in the ASO-MPL use case, with building density from left to right of 33%, 25%, and 11%.

C. Evaluation Results

In this section, we present our results on MPL-ASO AP selection using predicted LSF based on our REM model. In order to evaluate the AP selection performance in various propagation environments, we consider three city maps, namely, map 1, map 2, and map 3 with building density of 33%, 25%, and 11%, respectively, as illustrated in Figure 14. We assume that the 80 transmitter locations provided for each map in the *RadioMap3DSeer* dataset represent a CF-mMIMO network with 80 APs. To evaluate the performance of MPL-ASO with predicted LSF, we used a subset of the 80 APs, e.g., 70, to represent the set of active APs $\mathcal{A}_a$, which we use to train our u-net model $\mathcal{M}(\text{DAug}, \text{PxLoS}_f, \text{U-net}, \text{MSE})$, unless otherwise mentioned. Subsequently, we use this model to predict the LSF of the remaining 10 APs. We consider training and prediction using the full REM as well as a scattered LSF of selected outdoor locations. The cases where the number of APs to switch on is one, two, or three. In the following, we first define our performance metrics and then present our evaluation results.

1) Performance Metric: In this section, we use RMSE as a performance metric for the full REM prediction as well as for the scattered LSF prediction. However, unlike Section VI, where we calculate the RMSE of the whole output map, in this section, we only consider the selected outdoor locations in the RMSE calculation. The reason behind using a different RMSE calculation here is that in this section, we present cases where only a few outdoor locations are considered. In such cases calculating RMSE over the whole map might be misleading. Considering the output map $\mathbf{Y}_u$, we define the RMSE of an estimated map $\hat{\mathbf{Y}}_u$ as

$$\mathcal{E}_{\text{RMSE},u} = \sqrt{\frac{\sum_L \sum_N |\mathbf{Y}_u - \hat{\mathbf{Y}}_u|^2}{LN_u}}, \qquad (2)$$

where N_u is the number of the considered outdoor locations per map, which in the training phase represents the number of UEs, and L is the total number of maps used in the test set. To evaluate the AP selection performance, we conduct AP selection for the available outdoor user locations and calculate the error percentage across these locations as follows

$$\mathcal{E}_{\text{APsel}_u} = \frac{\sum_{n=1}^{N_u} \xi_n}{N_u}, \qquad (3)$$

where

$$\xi_n = \begin{cases} 0 & \text{if } \mathcal{A}_{sa} = \hat{\mathcal{A}}_{sa} \text{ at location } n \\ 1 & \text{otherwise} \end{cases} \qquad (4)$$
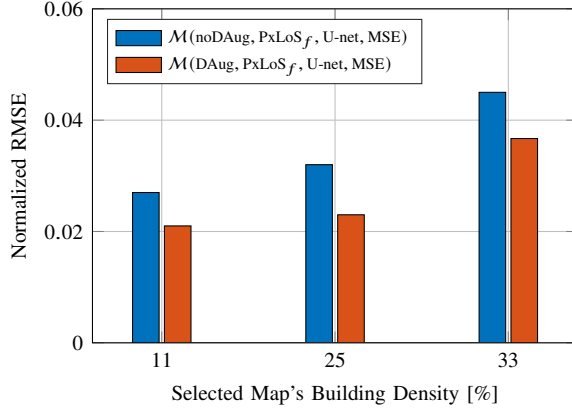
Fig. 15: Normalized RMSE of REM prediction of 10 off APs when training on a single map with 70 active APs, providing training data. Data augmentation is used to enrich the training data.
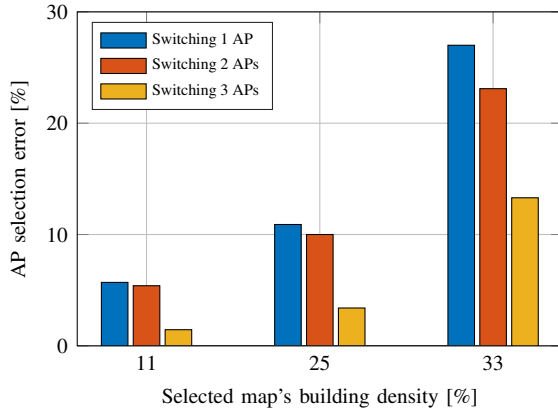

Fig. 16: AP selection error based on the predicted LSF of the off APs, calculated over all possible outdoor locations of the UE demanding extra SE, assuming full REMs of active APs are available.

with $\mathcal{A}_{sa}$ and $\hat{\mathcal{A}}_{sa}$ being the true and estimated set of APs to switch on.

2) Impact of Data Augmentation: In the MPL-ASO use case, we perform model training on a single map with various AP locations, resulting in a rather limited training set size compared to the results presented in Section VI. In order to enrich the training dataset and the model's ability to generalize, we employ the data augmentation method presented in Section V-D to obtain ×8 training dataset size. Figure 15 presents the $\mathcal{E}_{\mathrm{RMSE},u}$ of REM predicted for 10 off APs with 70 on APs used for model training (60 train and 10 validation). As shown in the figure, data augmentation outperforms the case without data augmentation for all three maps considered for the MPL-ASO use case. In particular, a gain in the $\mathcal{E}_{\mathrm{RMSE},u}$ of 0.009 for map 1 and map 2, and 0.006 for map 3 is obtained.

3) AP Selection Performance: The AP selection error is evaluated in Figure 16 for a scenario in which the full REM map of 70 active APs is available. In this case, we perform AP selection of the best one, two, and three APs from 10 sleep APs across all outdoor map locations. As shown in the figure, a maximum AP selection error of approximately 10% is achieved in map 2 and map 3. This means that for 90% of user locations, we are able to switch the optimal set
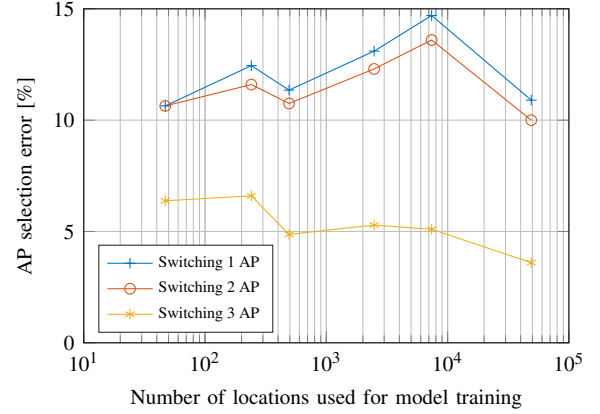

Fig. 17: AP selection error based on the predicted LSF of the off APs versus the number of locations/pixels used to train our u-net model for a map with a building density of 25%.

of APs without a need to switch any of the off APs to do channel gain measurements. In the case of map 1, which has a higher building density compared to map 2 and map 3, an AP selection error of 27% is obtained when only one extra AP is needed, whereas in case 3 extra APs are needed to meet the user's SE, the AP selection error goes down to 12%. This means that for the case where a single AP is needed at 15% of the locations, the second-best or third-best AP is turned on.

Since the assumption of having the full REM of active APs might be strict, in Figure 17, we present a case where only the LSF measurements at selected locations are available in map 2. Such measurements can be collected throughout the time window at which the active APs are on. We use the same locations to predict the LSF of sleep APs. These locations are randomly chosen to represent 15%, 5%, 1%, 0.5%, and 0.1% of the total number of pixels (256×256), which corresponds to the number of outdoor locations in map 2 of 7381, 2477, 493, 241, and 47, respectively. These different percentages of outdoor locations are chosen to asses our REM model with a wide range of outdoor locations, i.e., from tens to thousands of locations. The figure also presents the case where all outdoor locations in map 2 are considered, which corresponds to 48942 outdoor locations. As shown in the figure, an AP selection error lower than 15% is guaranteed for cases where one, two, or three extra APs are needed, regardless of the considered number of outdoor locations.

VIII. CONCLUSION

We presented an accurate time-efficient REM prediction framework based on u-net CNN. We investigated several data preprocessing steps and quantified their impact on the RMSE of predicted REMs. The presented preprocessing steps include three different approaches for fractional LoS maps calculation, building-density-based data split, data augmentation, as well as the u-net model loss function. We evaluated the performance of our proposed framework using the 3D city maps from the *RadioMap3DSeer* dataset and highlighted the performance gain and corresponding tradeoffs. In particular, the performance of our proposed framework has been evaluated in the context of *the 2023 IEEE ICASSP Signal Processing Grand Challenge,*

namely, the First Pathloss Radio Map Prediction Challenge. Our results have shown that the proposed framework provides a normalized average RMSE of 0.045 on the challenge's test set, with an average runtime of 14 milliseconds per map. Finally, a relevant CF-mMIMO use case is presented, in which we demonstrated that one could obviate consuming energy on large-scale fading measurements and rely on predicted REM instead to select which APs to switch on. In particular, we showed that by exploiting predicted REM an AP selection error of around 5% in case a UE's SE needs three extra APs is achieved.

REFERENCES

[1] Andreas Achtzehn, Janne Riihijärvi, Guilberth Martínez Vargas, Marina Petrova, and Petri Mähönen, "Improving coverage prediction for primary multi-transmitter networks operating in the tv whitespaces," in *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2012 9th Annual IEEE Communications Society Conference on*. IEEE, 2012, pp. 623–631.

[2] Shamik Sarkar, Aniqua Baset, Harsimran Singh, Phillip Smith, Neal Patwari, Sneha Kasera, Kurt Derr, and Samuel Ramirez, "LLOCUS: Learning-based Localization Using Crowdsourcing," in *Proceedings of the Twenty-First International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2020, pp. 201–210.

[3] E. Krijestorac, S. Hanna, and D. Cabric, "Spatial Signal Strength Prediction Using 3D Maps and Deep Learning," in *IEEE ICC*. IEEE, 2021.

[4] M Mahdi Azari et al., "Evolution of Non-Terrestrial Networks From 5G to 6G: A Survey," *IEEE communications surveys & tutorials*, 2022.

[5] Daniel Fabian Külzer, Sławomir Stańczak, and Mladen Botsov, "Cdi maps: Dynamic estimation of the radio environment for predictive resource allocation," in *2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2021, pp. 892–898.

[6] K. Rizk et al., "Two-Dimensional Ray-Tracing Modeling for Propagation Prediction in Microcellular Environments," *IEEE Trans on Veh. Tech.*, vol. 46, no. 2, 1997.

[7] 3GPP, "Study on channel model for frequencies from 0.5 to 100 ghz," document 3GPP TR 38.901, 2017.

[8] Eraldo Damosso, *Digital mobile radio towards future generation systems: COST action 231*, European Commission, 1999.

[9] Juha Meinilä, Pekka Kyösti, Tommi Jämsä, and Lassi Hentilä, "Winner ii channel models," *Radio Technologies and Concepts for IMT-Advanced*, pp. 39–92, 2009.

[10] R. Levie, Ç. Yapar, G. Kutyniok, and G. Caire, "RadioUNet: Fast Radio Map Estimation with Convolutional Neural Networks," *IEEE Trans. on Wir. Comm.*, vol. 20, no. 6, 2021.

[11] V. Ratnam et al., "FadeNet: Deep Learning-Mased mm-Wave Large-Scale Channel Fading Prediction and its Applications," *IEEE Access*, vol. 9, pp. 3278–3290, 2020.

[12] Shamik Sarkar, Dongning Guo, and Danijela Cabric, "ProSpire: Proactive Spatial Prediction of Radio Environment Using Deep Learning," *To appear in IEEE SECON*, 2023.

[13] "The First Pathloss Radio Map Prediction Challenge," https://radiomapchallenge.github.io/, 2023.

[14] Ç. Yapar et al., "Dataset of Pathloss and ToA Radio Maps with Localization Application," *IEEE Dataport*, 2022.

[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 2015, pp. 234–241.

[16] Enes Krijestorac, Hazem Sallouha, Shamik Sarkar, and Danijela Cabric, "Agile radio map prediction using deep learning," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–2.

[17] Guillem Femenias, Narjes Lassoued, and Felip Riera-Palou, "Access point switch on/off strategies for green cell-free massive mimo networking," *IEEE access*, vol. 8, pp. 21788–21803, 2020.

[18] Jan García-Morales, Guillem Femenias, and Felip Riera-Palou, "Energy-efficient access-point sleep-mode techniques for cell-free mmwave massive mimo networks with non-uniform spatial traffic density," *IEEE access*, vol. 8, pp. 137587–137605, 2020.

[19] Trinh Van Chien, Emil Björnson, and Erik G Larsson, "Joint power allocation and load balancing optimization for energy-efficient cell-free massive mimo networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6798–6812, 2020.

[20] Sung Joon Maeng, İsmail Güvenç, Mihail L Sichitiu, and Ozgur Ozdemir, "Out-of-Zone Signal Leakage Sensing in Radio Dynamic Zones," in *ICC 2022-IEEE International Conference on Communications*. IEEE, 2022, pp. 5579–5584.

[21] A. Chakraborty et al., "Specsense: Crowdsensing for Efficient Querying of Spectrum Occupancy," in *IEEE INFOCOM*, 2017.

[22] Yves Teganya and Daniel Romero, "Deep completion autoencoders for radio map estimation," *IEEE Transactions on Wireless Communications*, vol. 21, no. 3, pp. 1710–1724, 2021.

[23] Kehan Li, Chao Li, Baosheng Yu, Zhangchong Shen, Qiming Zhang, Shibo He, and Jiming Chen, "Model and transfer spatial-temporal knowledge for fine-grained radio map reconstruction," *IEEE Transactions on Cognitive Communications and Networking*, 2021.

[24] Theodore S Rappaport et al., *Wireless communications: principles and practice*, vol. 2, prentice hall PTR New Jersey, 1996.

[25] Andrea Goldsmith, *Wireless communications*, Cambridge university press, 2005.

[26] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou, "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems*, 2021.

[27] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org, 2017.

[28] R Hoppe, G Wölfle, and FM Landstorfer, "Fast 3d ray tracing for the planning of microcells by intelligent preprocessing of the database," in *3rd European Personal and Mobile Communications Conference (EPMCC)*. Citeseer, 1999.

[29] Reiner Hoppe, Gerd Wölfle, and Ulrich Jakobus, "Wave propagation and radio network planning software winprop added to the electromagnetic solver package feko," in *2017 International Applied Computational Electromagnetics Society Symposium-Italy (ACES)*. IEEE, 2017, pp. 1–2.

[30] "A Rasterizing Algorithm for Drawing Curves," http://members.chello.at/easyfilter/Bresenham.pdf, 2012.

[31] Ryosuke Okuta, Yuya Unno, Daisuke Nishino, Shohei Hido, and Crissman Loomis, "Cupy: A numpy-compatible library for nvidia gpu calculations," in *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.

[32] Ju-Hyung Lee, Joohan Lee, Seon-Ho Lee, and Andreas F Molisch, "Pmnet: Large-scale channel prediction system for icassp 2023 first pathloss radio map prediction challenge," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–2.

[33] Solomon Kullback and Richard A Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[34] Özlem Tugfe Demir, Emil Björnson, Luca Sanguinetti, et al., "Foundations of user-centric cell-free massive mimo," *Foundations and Trends® in Signal Processing*, vol. 14, no. 3-4, pp. 162–472, 2021.

[35] Giovanni Interdonato, Emil Björnson, Hien Quoc Ngo, Pål Frenger, and Erik G Larsson, "Ubiquitous cell-free massive mimo communications," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, pp. 1–13, 2019.

[36] Shuaifei Chen, Jiayi Zhang, Jing Zhang, Emil Björnson, and Bo Ai, "A survey on user-centric cell-free massive mimo systems," *Digital Communications and Networks*, vol. 8, no. 5, pp. 695–719, 2022.